基于pdf的电子印章应用的设计与实现

**团队名称：黑槿花**

**团队人员：姚铭业**

**王华彬**

**杜康镇**

**滕宇生**

**雷富强**

**学 院：计算机与信息安全学院**

**指导老师：张瑞霞**

**日期：2018年12月20日**

# 摘要

在中国，印章在现实生活中是重要的征信工具，可以用来证明身份、信义、关系和合同有效性等，广泛应用于行政机关、医院、企业、档案馆等单位。从日常的购物发票上的发票专用章，到行政机关的发布公告的使用的公章，印章在日常生活中随处可见。看一份通知或合同是否有效，最直观的方法就是查看其是否有加盖印章。凡是需要信息载体具备可信性，加盖印章是最直接、有效的办法。

随着互联网技术的发展，无纸化办公在我国开始逐渐普及，通过使用PDF文档、Word文档等记录文字图片信息，通过邮箱发送、网络广播等方式进行文件的收发、信息的传播，实现日常办公的无纸化。同样的合同、通知、公告等需要具备真实性、有效性的也可以通过上述方式进行推送，而想要证明其有效、真实，就需要使用到印章，电子印章就是为解决这一问题而生的。

而在我国互联网的高速发展的同时，我国的操作系统的开发也没有停下其脚步，为了能有稳定、安全的操作系统可以使用，我国有包括武汉深之度科技有限公司、上海中标软件有限公司等多家公司进行国产操作系统的研发。我国的操作系统是基于Linux平台进行开发的，典型的有Deepin、红旗Linux、中标麒麟Linux。这些操作系统是能够满足用户的日常工作学习需要的，唯一的缺点是缺乏足够的商业软件支持、软件生态还不够完善。可以说，中国在这一领域还有很大的发展空间。

我们的这款PDF电子印章应用软件，主要用于解决在Linux平台没有电子印章盖章软件的这一实际问题，同时也是为了完善我国Linux平台的软件生态，使其在电子印章软件这一块拥有能够替代Windows和Mac平台相应软件的能力，并加强我国操作系统在无纸化办公这一领域的竞争力。

关键字：印章、电子印章、无纸化办公、Linux、国产操作系统

目录

[1. 绪论 1](#_Toc26678990)

[1.1 背景和意义 1](#_Toc26678991)

[1.1.1印章的概念、作用与分类 1](#_Toc26678992)

[1.1.2印章的历史 1](#_Toc26678993)

[1.1.3便携式文档格式 2](#_Toc26678994)

[1.1.4基于Linux平台的面型PDF的电子印章系统 2](#_Toc26678995)

[1.2 发展现状分析 3](#_Toc26678996)

[1.3 可行性分析 3](#_Toc26678997)

[1.3.1经济 3](#_Toc26678998)

[1.3.2技术 3](#_Toc26678999)

[1.3.3政策 3](#_Toc26679000)

[1.4 本文主要内容 3](#_Toc26679001)

[2. 需求分析 4](#_Toc26679002)

[2.1 用户及业务需求 4](#_Toc26679003)

[2.1.1功能需求 5](#_Toc26679004)

[2.2 功能简介 6](#_Toc26679005)

[2.2.1客户端电子印章应用系统简介 6](#_Toc26679006)

[2.2.2软件网站简介 6](#_Toc26679007)

[2.2.3信息管理系统简介 6](#_Toc26679008)

[2.3 层次方框图 7](#_Toc26679009)

[2.4 数据流图（DFD） 8](#_Toc26679010)

[3. 总体设计 10](#_Toc26679011)

[3.1 模块层次图 10](#_Toc26679012)

[3.2 客户端接口 10](#_Toc26679013)

[3.2.1打开 10](#_Toc26679014)

[3.2.2关闭 11](#_Toc26679015)

[3.2.3页面缩放 11](#_Toc26679016)

[3.2.4页面跳转 11](#_Toc26679017)

[3.2.5制作电子印章 12](#_Toc26679018)

[3.2.6盖章 12](#_Toc26679019)

[3.2.7验证 12](#_Toc26679020)

[3.2.8撤章 13](#_Toc26679021)

[3.2.9登陆 13](#_Toc26679022)

[3.3 网页端接口 13](#_Toc26679023)

[3.4 系统输入和输出 21](#_Toc26679024)

[3.5 数据库设计 22](#_Toc26679025)

[3.5.1数据字典 22](#_Toc26679026)

[3.5.2数据库E-R图 26](#_Toc26679027)

[4. 详细设计 27](#_Toc26679028)

[4.1 PDF文档操作模块设计 29](#_Toc26679029)

[4.1.1打开PDF文档 29](#_Toc26679030)

[4.1.2页面大小调整 29](#_Toc26679031)

[4.1.3页面跳转 30](#_Toc26679032)

[4.1.4保存文档 31](#_Toc26679033)

[4.2电子印章操作 31](#_Toc26679034)

[4.2.1制作电子印章 31](#_Toc26679035)

[4.2.2盖章 32](#_Toc26679036)

[4.2.3撤章 33](#_Toc26679037)

[4.2.4验证 34](#_Toc26679038)

[4.3 客户端网络相关操作 35](#_Toc26679039)

[4.3.1登陆 35](#_Toc26679040)

[4.3.2网络传输 36](#_Toc26679041)

[4.4 网页端具体设计 37](#_Toc26679042)

[4.4.1用户公司信息及证书信息上传 37](#_Toc26679043)

[4.4.2管理员用户信息管理 38](#_Toc26679044)

[4.4.3管理员审核公司信息 39](#_Toc26679045)

[4.4.4管理员审核证书信息 41](#_Toc26679046)

[5. 系统实现 42](#_Toc26679047)

[5.1 PDF文档操作模块 42](#_Toc26679048)

[5.1.1思路 42](#_Toc26679049)

[5.1.2相关类及其描述 43](#_Toc26679050)

[5.1.3核心代码 53](#_Toc26679051)

[5.2 电子印章操作模块 55](#_Toc26679052)

[5.2.1思路 55](#_Toc26679053)

[5.2.2相关类及其描述 59](#_Toc26679054)

[5.2.3核心代码 73](#_Toc26679055)

[5.3 网络相关操作模块设计 77](#_Toc26679056)

[5.3.1思路 77](#_Toc26679057)

[5.3.2相关类及其描述 77](#_Toc26679058)

[5.4 网页端客户上传信息模块 78](#_Toc26679059)

[5.4.1思路 78](#_Toc26679060)

[5.4.2相关类及描述 78](#_Toc26679061)

[5.4.3核心代码 78](#_Toc26679062)

[5.5 网页端管理员审核信息模块 82](#_Toc26679063)

[5.5.1思路 82](#_Toc26679064)

[5.5.2相关类及描述 82](#_Toc26679065)

[5.5.3核心代码 82](#_Toc26679066)

[6. 软件使用测试 86](#_Toc26679067)

[6.1 PDF文档操作模块 87](#_Toc26679068)

[6.1.1打开文件 87](#_Toc26679069)

[6.2 电子印章操作模块 87](#_Toc26679070)

[6.2.1盖章 87](#_Toc26679071)

[6.2.2撤章 88](#_Toc26679072)

[6.2.3制章 89](#_Toc26679073)

[6.2.4验证 90](#_Toc26679074)

[6.3服务器端相关模块 92](#_Toc26679075)

[7. 结束语 96](#_Toc26679076)

[7.1总结 96](#_Toc26679077)

[7.1.1已完成的功能模块 96](#_Toc26679078)

[7.1.2使用价值 96](#_Toc26679079)

[7.2展望 96](#_Toc26679080)

[7.2.1未完成的功能模块 96](#_Toc26679081)

[7.2.2下一步的打算 97](#_Toc26679082)

[参考文献 99](#_Toc26679083)

# 绪论

## 1.1 背景和意义

### 1.1.1印章的概念、作用与分类

印章，又称图章。是指将文字、符号和图形按照一定的艺术风格或规范格式刻制、铸造在物体上，形成的以改制图形和文字标记为使用方式的标志性信物和艺术品。是用作文件上表示鉴定和签署的文具。一般印章会沾上颜料再印上，不沾颜料的印章，印上后平面会带有凹凸的叫钢印。有些是用于蜡或火漆上。制作的主要材料有金属、木头、玉石、石头等。通常与特定的身份绑定，是实现身份认证的工具。且具有一定的法律效应。

签章，本文中指印章盖章后的留下的印记。

电子印章，是合法的数字化印章与数字证书绑定的，用其私钥进行数字签名的权威性电子文件。它是一种数据结构，该结构模拟现实中电子印章的功能。主要包含了用户身份、印章信息、公开密钥。

电子签章，与我们通常理解的物理印章在纸上或其他媒介上盖章后留下的印记对应。本文中电子签章也是一种数据结构，它含有对应电子印章的基础信息、文档信息摘要等信息。具有自我保护能力。

我国于2005年4月1日正式实施《电子签名法》，对电子印章进行了详细的定义。电子印章是电子形式的数据，是用来识别签名人的身份和表明签名人认可该数据电文内容的数据。与物理印章不同的是，电子印章还具有防篡改、安全、保密等特性。［３］

我国规定电子印章分为圆形、椭圆形和方形。其中国家行政机关和企业、事业单位、社会团体、学校及其所属职能机构、国有企业、国营股份制企业、股份有限公司、集体企业、私营企业、有限责任公司、工会、工商企业合同专用章、业务专用章使用圆形印章。中外合资、外商独资经营企业、发票专用章、报关专用章使用椭圆形印章。财务专用章、法人章使用方形印章。

### 1.1.2印章的历史

印章是我国特有的历史文化产物，古代主要用于身份凭证和行驶权力的象征。它的起源，是由于社会生活的需要，早在商周时期，印章就已经产生。在印章的发展过程中，印章曾出现两次发展高潮期。第一次是在秦汉时期，第二次是在明清时期。秦统一中国后，在加强中央集权、吸收各地文化。反映在印章上，就是制度的等级性，文字上的统一性以及风格上的朴实性。为了体现王权的尊贵，皇帝使用的印章被称为“玺”，以玉制作；臣子使用的印章叫“玉”，用金、银、铜等制作。两汉时期继承和完备了秦朝时的官印制度，使印章的规格、[印钮](https://baike.sogou.com/v70010684.htm?ch=ww.wap.chain.baike)的形制及印文的形态更趋统一，而且形成了汉印特有的博大精深、气象万千的艺术风貌。汉印的成就，主要可归纳于以下几点：制度完备、种类繁多、文字规范、章法自然、记忆精湛。

明清时期的印章逐渐走向民用，并且不断发展，最终印章不但具有实用价值，还是一种艺术品。近代的印章的作用主要继承于明清时期的应用。

### 1.1.3便携式文档格式

便携式文档格式（Portable Document Format简称PDF），是由Adobe Systems用于与应用程序、操作系统、硬件无关的方式进行文件交换所发展出来的文件格式。现在由国际化标准组织（ISO）来维护。PDF以PostScript语言图像模型为基础，无论在哪种打印机上都可保证精确的颜色和准确的打印效果，及PDF会完美的还原原文档的每一个字符、颜色以及图像。PDF文档遵守PDF格式的文件，所以PDF文档与平台无关，也就是说无论是Windows平台，Linux还是Android。所有平台上的PDF文档格式都相同，解析器只需要按照PDF的渲染格式来渲染PDF就好，也就是说无论什么平台上的PDF文档所看到的结果都一样。［４］

### 1.1.4基于Linux平台的面型PDF的电子印章系统

传统的印章主要用于对文档进行身份的验证，传统印章经过几千年的发展已经有了一套非常完善的印章真伪验证体系。如今，计算机慢慢融入人们的生活，政府机关和社会上不可避免的开始使用电子文档作为主要信息传输媒介。而电子文档的安全验证就成了不可避免的问题。传统印章无法在电子文档上盖章，这一现状限制了电子文档的发展。而PDF文档能够很好的解决这一问题，首先PDF文档在所有的平台上都能得出一样的渲染结果，这样电子印章的外观在所有的PDF文档上都能保持外观的一致性。所以PDF文档适合用来作为电子文档的载体。

电子印章在我国的《电子签名法》颁布后正式具有法律效应。而电子信息可以进行各种加密来保证数字签名的安全性。使得电子印章的安全性远高于传统印章。在用户选择对文档进行盖章后电子印章取PDF文档的部分进行消息摘要，如果有人在中途修改了PDF文档，在验证时能够通过公钥检查出来。且电子印章的使用方式与传统印章相似。符合人们对传统印章的使用习惯。

Linux操作系统是一款免费使用和自由传播的类Unix操作系统。Linux存在许多不同的版本，但是他们都有相同的内核。Linux可以安装到不同的计算机硬件设备中，严格说Linux是指Linux内核。我国也有很多不同版本的Linux操作系统。这对于国内的政府以及企业来说安全性要比国外的操作系统高很多。因此开发一款基于Linux平台的面型PDF文档的电子印章系统显得很有必要。［１］［２］

## 1.2发展现状分析

对于本产品的发展现状分析，目前能实现盖章功能的软件，主要是Adobe公司的Acrobat，foxit的pdf编辑器，以及NTKO文档管理系统，但这三个软件都只面向Windows、Mac、iOS和安卓平台，在Linux平台，还没有电子印章软件，所以本产品近期在Linux平台上，具有巨大优势，发展前景可观，不过，值得注意的是以目前的软件发展趋势，在不久之后就会有大量同类型软件进入相同平台，所以未来持久发展不可以太过乐观。

## 1.3可行性分析

### 1.3.1经济

从经济方面来考量，本程序开发所用软件基本为开源软件，所以经济方面可行。

### 1.3.2技术

从技术层面来说，开发人员的技术层面基本达到要求，所以技术层面可行。

### 1.3.3政策

从政策方面来讲，本软件符合国家政策法律，和相关的基本条例，政策方面可行。

## 1.4本文主要内容

本文主要是对基于Linux的面向pdf的电子印章应用系统的描述、分析、设计进行说明。通过本文可了解到整个系统的设计过程，了解到其实现的功能以及在市场上的应用价值。

# 需求分析

## 2.1 用户及业务需求

对于本系统用户所面临的主要问题是如何验证电子印章的有效性以及用户如何确保PDF文档的完整性。即在传输的过程中PDF文档以及电子印章没有被任何人以任何方式修改。PDF文档的完整性是指在用户盖章完成后电子印章没有任何的修改。并且该系统需要认证盖章者的身份信息。即确保盖章者的身份是可信的，盖章者不能否认盖章行为以及盖章文件的有效性。

在本系统中，电子印章与数字证书进行关联以及服务器保存相关印章信息来保证电子印章的有效性与安全性。主要关联方式为把数字证书认为是电子印章的一部分。通过数字证书来认证盖章人的身份。通过电子印章中的信息摘要来验证PDF文档的完整性。通过数据库保存的信息来保证用户的文档内容以及电子印章没有被人修改。

本系统中的电子印章盖章方式是模仿现实中的传统印章的盖章方式，即取章、盖章。电子印章盖章方式为用户点击盖章后选择在PDF中的盖章位置。随输入电子印章的密码，通过盖章模块将电子印章放入PDF文档对应的位置。

盖章者盖章的目的使用一个电子数据来认证PDF文档，并证明PDF文档的有效性。所以盖章人需要给出自己的电子印章已得到一份以及盖章完毕的PDF文档。同时出于现实情况还需要盖章文件进行撤章操作。这是为了避免用户对某一PDF文档进行盖章后发现选择的电子印章有误，用户可以随时取消自己盖的电子印章。同时由于电子印章的普及并不是很好，存在大量的用户没有电子印章。所以也需要为用户提供制作电子印章的功能，用户只需要提供相应的电子印章图片以及用于验证的ＣＡ证书和电子印章密码即可生成电子印章。

认证者需要PDF文档的盖章人身份的有效性进行验证，以及对PDF文档的完整性进行验证。只有这样认证者才能相信PDF文档内容的真实性。用户想要验证盖章人的信息只需要选择批量验证，然后再选择相应的电子印章，随后显示电子印章的验证信息。用户想要验证PDF文档的有效性只需要点击验证证书查看证书具体信息是否和盖章者一致以及PDF文档的信息摘要是否与盖章时不同即可。

CA证书与电子印章相联系来管理电子印章最早出自王延彬［５］《基于CA的电子印章系统的研究与实现》，通过电子印章管理后台系统的概念来对电子印章进行管理，本文也采用了电子印章管理系统的结构来解决电子印章管理的问题。

### 2.1.1功能需求



图2.1.1功能需求用例图

PDF文档操作用例是指系统对PDF文档的一切操作，包括打开PDF文档，浏览，关闭，修改等。该用例是除制作电子印章用例外一切用例的基础。只有先打开了PDF文档后才能开始其他功能。

制作电子印章的参与者是一切用户，所有的用户都可以根据自己的需求选择是否制作电子印章。当用户选择制作电子印章是，在制章页面输入电子印章图片，CA证书，用户密码等来设置自己的电子印章，随后返回制章情况给用户。结束该用例。

盖章与撤章用例的参与者是盖章者，用例要实现的功能是对PDF文档进行盖章或撤章。当用户选择盖章并在PDF文档上选择盖章位置后让盖章人选择要使用的电子印章并输入密码，密码正确则在对应位置盖上电子印章。密码验证失败则提示用户密码错误。当用户选择撤章是先获取该PDF文档上的电子印章，并在给用户提示，用户可以悬着PDF文档上有的电子印章。用户选择电子印章后给出该印章的预览，以免用户撤销还需要用的电子印章。选择印章后用户输入电子印章密码，验证成功后撤销电子印章、更新PDF文档并给用户成功提示。密码验证失败给用户失败提示。

验证用例的用户是验证者。验证者需要验证电子印章的有效性以及PDF文档的完整性。当验证者选择验证电子印章功能后系统先自动验证PDF文档内的电子印章是否失效。并给验证者相应的提示。并给出电子印章选择，用户可以选择自己想要验证的电子印章。选择后会给出电子印章的验证信息。验证者可以通过信息对比来查看电子印章的有效性。验证者想要对PDF文档的有效性进行验证时选择验证证书功能。系统会给出电子印章对应的CA证书，用户选择证书后给出证书的验证结果。同时，还能查看印章记录在服务器端的相关信息

## 2.2 功能简介

本软件分为三大部分：第一部分是客户端电子印章应用系统，第二部分是软件网站，第三部分是信息管理系统。

### 2.2.1客户端电子印章应用系统简介

客户端的电子印章应用系统是本系统的一个核心部分，主要作用有浏览PDF、制作电子印章、对PDF盖章、撤章、验证由本系统所盖的PDF电子印章。

主要功能包括以下几个方面：

1. 基于poppler实现的PDF浏览器功能，显示用户打开的文件；
2. 将JPG格式和JPEG格式的图片制作成电子印章；
3. 对PDF文件进行盖章和将盖章信息提交到服务器端保存；
4. 对PDF文件中的印章进行撤除印章操作，不过只能撤还未将信息提交到服务器端保存的已盖章PDF文件，信息一旦提交就不允许进行撤章；
5. 验证由本软件盖在PDF文件中的电子印章，并从服务器获取该盖章文件的信息。

### 2.2.2软件网站简介

软件的配套网站用于为客户提供账号注册、软件下载、企业信息提交认证服务、证书提交认证服务。

### 2.2.3信息管理系统简介

该模块用于管理用户和用户提交的信息，主要作用是进行用户管理、用户企业信息认证、用户证书认证。该模块由软件的运营商使用。用户的信息存储在云数据库中，提高安全性。

## 2.3 层次方框图

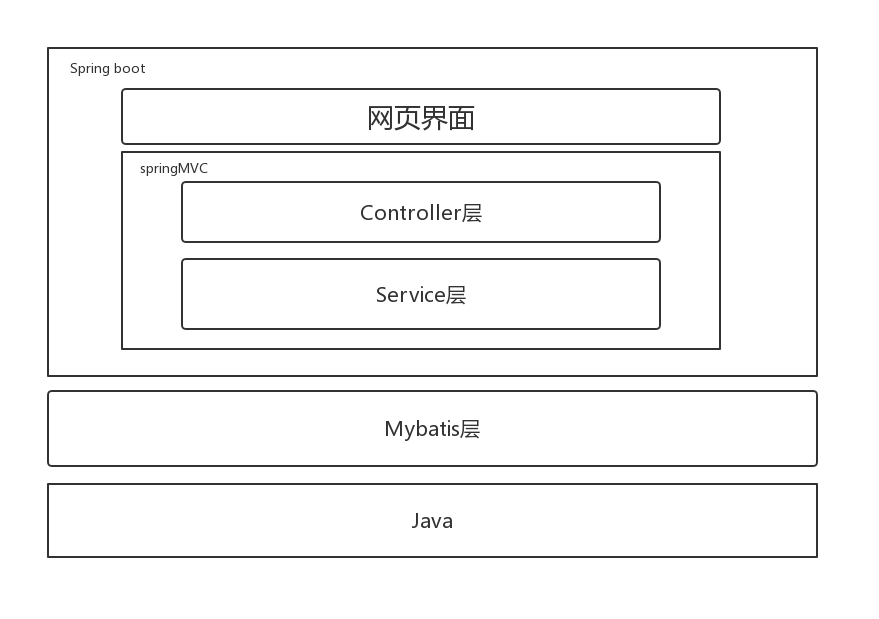


图2.3-1 服务器端层次方框图

服务器端采用SSM架构，同时处理来自网页端和客户端通过接口传来的信息。网页前端采用Vue和ElementUI实现，数据库采用云数据库，通过Mybatis进行管理。

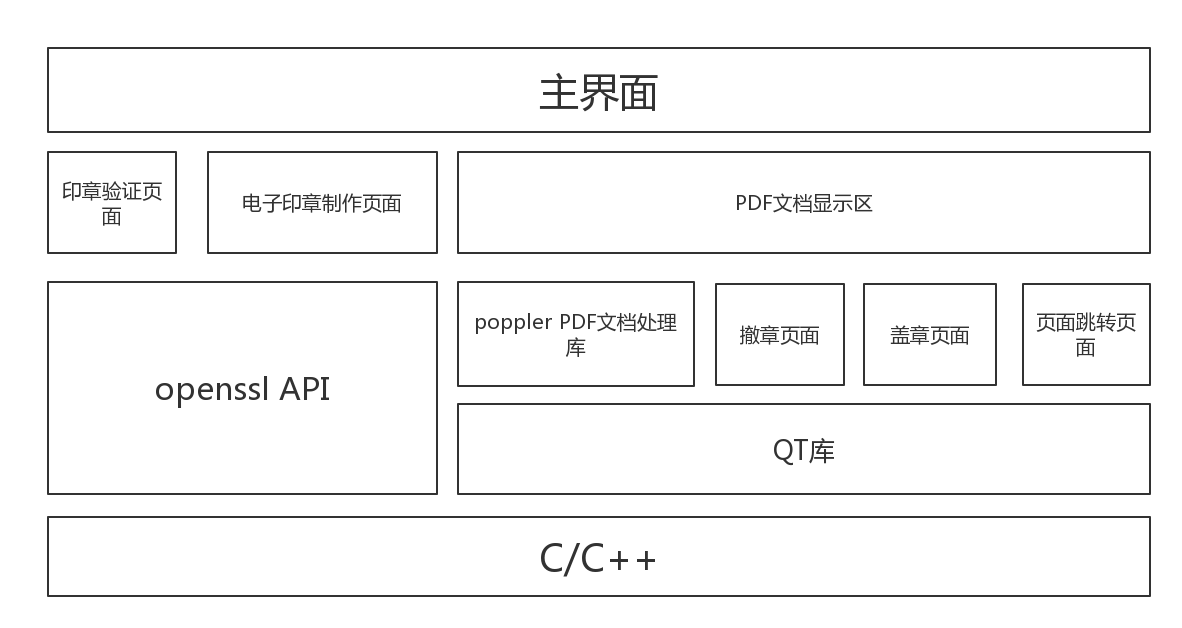


图2.3-2 客户端层次方框图

该图为总体的层次方框图，描述了软件的总体层次结构，以及各部分所依赖的技术。电子印章的制作与验证通过openssl技术实现，PDF文档的显示通过poppler库进行实现，撤章、盖章、页面跳转，则是通过QT库的文件读写操作和数据结构实现。这些库都是通过C/C++实现的。

图2.3-3 印章操作模块的局部层次方框图

以上的两个层次方框图，是对总体层次方框图的局部功能，即PDF文档操作功能和印章操作功能，进行细化。描述了它们所涵盖的操作。

## 2.4 数据流图（DFD）

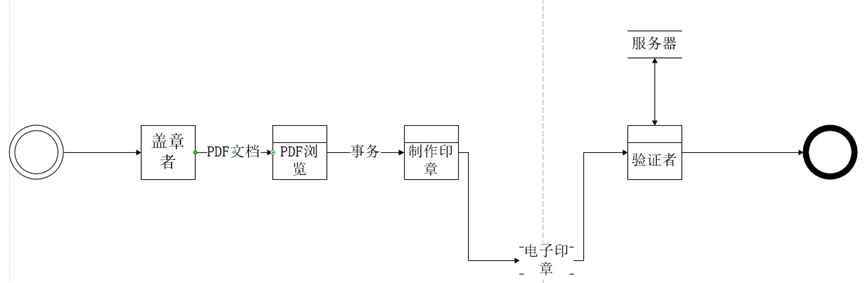


图2.4-1制作电子印章数据流图

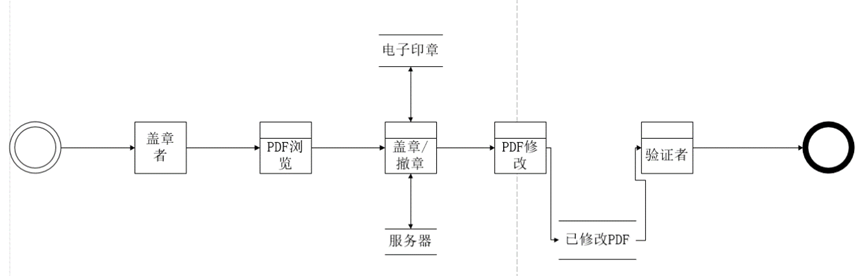


图2.4-2盖章／撤章数据流图

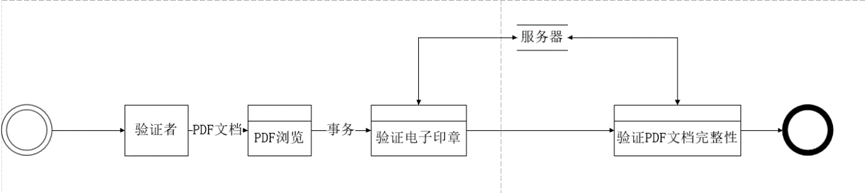


图2.4-3验证数据流图

以上三幅数据流图用于描述了电子印章的制作、盖章、撤章、验证四种操作所必须完成的逻辑功能。

# 总体设计

## 3.1 模块层次图

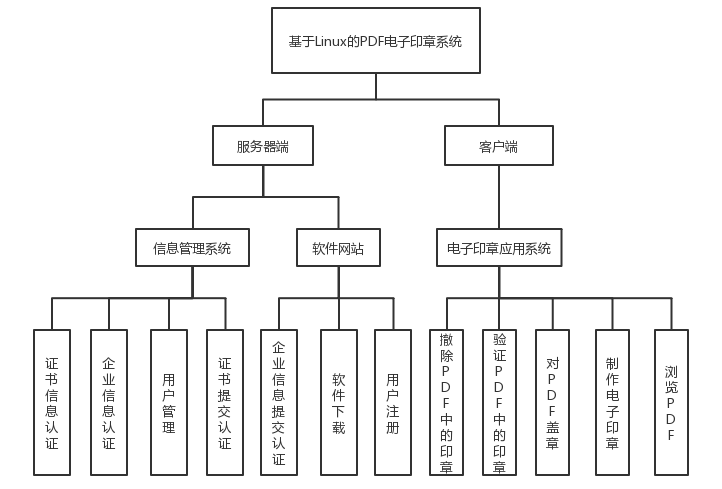


图3.1 程序模块层次图

## 3.2 客户端接口

PDF文档操作模块的接口主要3种，第一种是PDF文档的打开和关闭，第二种是PDF文档页面的缩放，第三种是PDF文档页面跳转。PDF印章操作模块有4种，第一种是制作电子印章，第二种是盖章，第三种是验证印章，第四种是撤章。

### 3.2.1打开

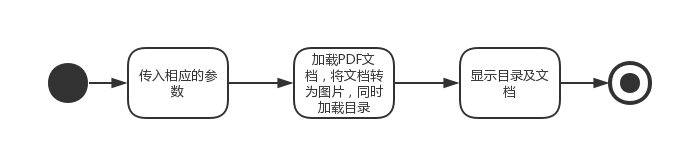
打开PDF文档需要的方法是BusinessModel类的openFile方法，共需要传入6个参数，传入的参数分别是：1、PDF文件地址，类型为QString；2、存储页面图片的数组，类型为QVector<QImage>\*\*；3、页面显示区域，类型为QScrollArea\*；4、页码存储，类型为int\*；5、目录，类型为QListWidget\*；6、图片缩放比例，类型为double。无返回值。

图3.2.1打开文件流程

### 3.2.2关闭

关闭的方法是BusinessModel类的freeResource方法，释放已缓存的PDF文档信息，无参数无返回；QListWidgetItem的takeItem方法，清除目录，参数为int型，返回为QListWidgetItem\*；QWidget的tekeWidget方法清空页面。

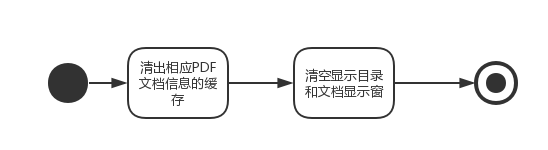


图3.2.2关闭图片流程

### 3.2.3页面缩放

页面缩放对应的方法是BusinessModel类的zoom方法，参数有4个，分别是页面图片缓存信息，类型为QVector<QImage>\*\*；当前页码，类型为int；放大倍数，类型为double；页面显示区域，类型为QScrollArea\*。无参数返回。根据需要实现放大还是缩小传入相应的缩放倍数即可。

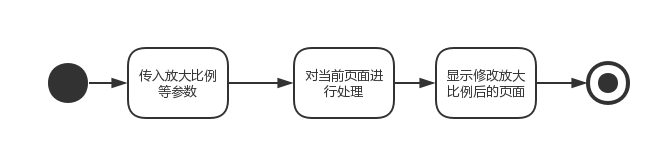


图3.2.3页面缩放流程

### 3.2.4页面跳转

页面跳转对应的方法为BusinessModel类的jumpPage方法，参数有4个，当前页码，类型为int；页面图片缓存信息，类型为QVector<QImage>\*\*；页面显示区域，类型为QScrollArea\*；PDF文件地址，类型为QString。根据需求可调用页面跳转页面，通过输入页码作为参数，实现页面跳转。

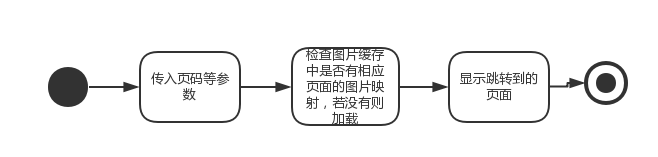


图3.2.4页面跳转流程

### 3.2.5制作电子印章

制作电子印章对应的方法是BusinessModel类的makeSeal方法，无参数无返回。调用该方法后会弹出制作电子印章的界面，需要用户手动选择证书、图片，手动输入印章的各项信息，作为参数，所有参数均为QString类型。

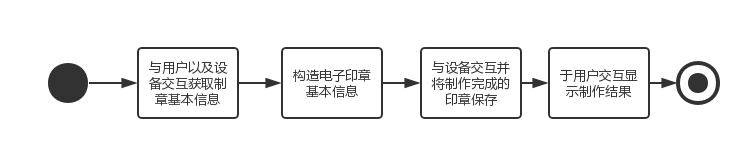


图3.2.5制章流程

### 3.2.6盖章

与盖章对应的方法是BusinessModel类的seal方法，这个函数接收八个参数，第一个是页面图片缓存信息，类型为QVector<QImage>\*\*；第二个是页面显示区域，类型为QScrollArea\*；第三第四个是PDF文档的坐标位置都是整形类型；第五个是PDF文档的位置，是QString类型；第六个是电子印章的位置，是QString类型；第七个是用户输入的密码，是QString类型；第八个是盖章的页码是整形类型，无返回值。

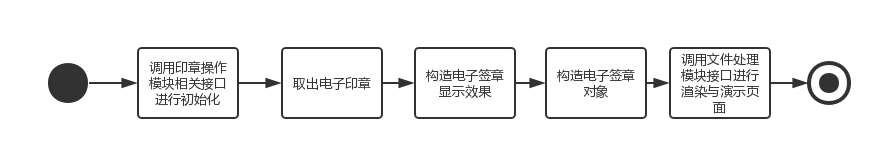


图3.2.6盖章流程

### 3.2.7验证

与验证对应的函数是batchvalidation，该函数接收一个QVector<QVector<QByteArry>>类型和一个字符串类型参数，无传出参数。

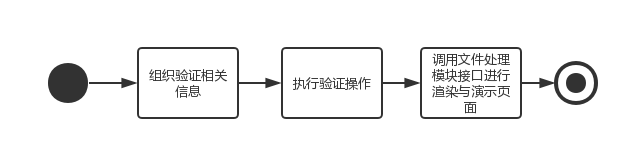


图3.2.7验证流程

### 3.2.8撤章

与撤章对应的方法是BusinessModel类的unseal方法，该方法的参数有3个，第一个是页面图片缓存信息，类型为QVector<QImage>\*\*；第二个是页面显示区域，类型为QScrollArea\*；第三个参数是当前页面，类型为int型。无返回值。

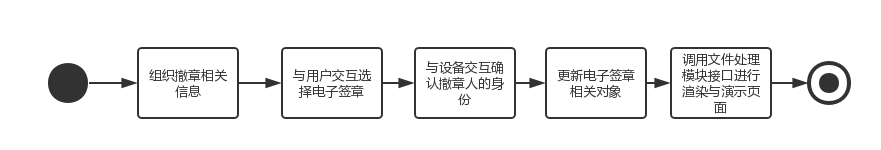


图3.2.8撤章流程

### 3.2.9登陆

与登陆对应的方法为LoginDialog类中的相关方法，传入的参数为用户名和用户密码。登陆成功会返回用户信息类型为QJsonObject。



图3.2.9登陆流程

## 网页端接口

网页部分接口主要是前后端之间的接口，按照具体的功能进行划分，如下所示：

登录

接口功能：完成用户身份认证。登陆成功后，默认服务端安全认证框架可获取到已登录用户的信息，以后的请求不需要再附加用户信息。

访问路径：/user/login

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| username | 用户名 | String | \* | 无 |
| password | 密码 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

注册

接口功能：注册用户。

访问路径：/user/register

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| username | 用户名 | String | \* | 无 |
| password | 密码 | String | \* | 无 |
| email | 邮箱 | String | \* | 无 |
| cellphone | 用户手机号 | Int | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

登出

功能：退出登录

访问路径：/user/logout

请求方式：GET

参数说明：无

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

上传企业信息

接口功能：上传用户企业相关信息。

访问路径：/submit/companyInfo

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| emile | 用户邮箱 | String | \* | 无 |
| company | 公司名称 | String | \* | 无 |
| creditcode | 企业信用代码 | String | \* | 无 |
| verifyinfo | 验证信息 | Int | \* | 无 |
| filename | 营业执照名字 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

上传证书信息

接口功能：上传用户证书相关信息。

访问路径：/submit/certifyInfo

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| id | 用户id | String | \* | 无 |
| email | 用户邮箱 | String | \* | 无 |
| certificate | 证书存储位置 | String | \* | 无 |
| filetype | 证书类型 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

加载用户信息

接口功能：获取已登录用户的基础信息。

访问路径：/user/info

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

注销用户

接口功能：管理员删除已注册用户。

访问路径：/user/delete

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

修改密码

接口功能：修改用户密码。

访问路径：/user/ changePassword

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |
| oldpassword | 旧密码 | String | \* | 无 |
| newpassword | 新密码 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

创建新用户

接口功能：添加新用户。

访问路径：/user/creat

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |
| username | 用户名 | String | \* | 无 |
| password | 密码 | String | \* | 无 |
| department | 公司名 | String | \* | 无 |
| cellphone | 电话号码 | String | \* | 无 |
| power | 权限 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

重置密码

接口功能：管理员重置普通用户密码。

访问路径：/user/resetPassword

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

修改用户信息

接口功能：管理员修改普通用户信息。

访问路径：/user/changeUser

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |
| username | 用户名 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

加载用户信息

接口功能：管理员查看用户信息。

访问路径：/user/employees

请求方式：GET

参数说明：无

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

加载公司信息审核表

接口功能：管理员审核公司信息信息。

访问路径：/vertify/companytable

请求方式：GET

参数说明：无

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

加载证书信息表

接口功能：管理员查看证书审核表。

访问路径：/vertify/verifytable

请求方式：GET

参数说明：无

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

公司审核信息详情

接口功能：管理员审核公司信息。

访问路径：/vertify/companyinfo

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |
| company | 公司名 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

公司信息通过审核

接口功能：管理员审核公司信息通过。

访问路径：/vertify/confirm

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |
| state | 认证信息 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

公司信息驳回审核

接口功能：管理员审核公司信息驳回。

访问路径：/vertify/companyRejec

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |
| state | 认证信息 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

证书信息通过审核

接口功能：管理员审核证书信息通过。

访问路径：/submit/confirm

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |
| state | 认证信息 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

证书信息驳回审核

接口功能：管理员审核证书信息驳回。

访问路径：/submit/ certifyReject

请求方式：GET

参数说明：

| 参数名称 | 参数含义 | 类型 | 是否必填 | 缺省值 |
| --- | --- | --- | --- | --- |
| email | 用户邮箱 | String | \* | 无 |
| state | 认证信息 | String | \* | 无 |

结果说明：

| 参数名称 | 类型 | 含义 | 说明 |
| --- | --- | --- | --- |
| message | String | 描述信息 |  |
| data | Object | 信息对象 |  |

## 系统输入和输出

用户打开软件后，在打开PDF文档前，除打开文档按钮、菜单中的打开选项以及电子印章制作按钮外，点击区域的选项都会弹出未打开PDF文档的提示窗。

打开文件后，文档显示框会显示文档图片，左侧的选项列表会逐渐加载PDF文档的缩略图，点击缩略图可进行页面跳转。

当点击各类页面跳转按钮时，会实现翻页等效果，也可通过点击当前的页码弹出页面跳转界面实现页面的跳转。当点击各类缩放按钮时，页面会进行缩放。

当点击盖章按钮时，按钮会凹下去，在文档显示区域中想要盖章的地方点击鼠标左键，会弹出选择印章和输入密码的提示框，输入密码并确认后，印章就被盖在PDF页面上了。

当点击验证按钮时，会弹出印章验证界面，若无印章就是空的，当有印章的时候，点击相应印章会显示印章的信息，点击查看签证证书按钮，会弹出证书信息界面，选择相应证书就能查看相应的信息，同理，查看时间戳证书也是一样的效果。

当点击印章制作按钮时，会弹出印章制作界面，填完参数后点击确定即可制作印章，只有通过这种方式做出来的印章才能盖章。

当点击撤章按钮后，会弹出撤章界面，若无印章就是空的，有印章时，通过点击列表中的印章，可查看印章在文档中的编号、所在的页码以及预览图，点击确定后会弹出密码输入框要求输入密码，输入正确密码点确定后，被撤掉的印章就会从列表中消失，关闭撤章界面后，文档显示区域的印章图片也随之消失。

用户可以在官网上查看本系统的相关信息，可以使用官网注册/登录网站使用网站的具体功能，具体功能有公司信息提交以及证书信息提交，提交后可等待管理员的认证通过或者驳回。也可进行注销

管理员可以对用户进行管理，包括修改用户的信息，查看所有用户，重置用户的密码以及修改自己的密码，也可以创建新用户。以及查看公司信息审核表和证书信息审核表，对于公司信息审核表需要管理员点击公司详情查看具体内容后进行通过或者驳回，证书信息审核需要管理员将证书信息下载下来然后对其进行审核。

## 数据库设计

### 3.5.1数据字典

表3.5.1certificate表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **空** | **默认** | **注释** |
| id (主键) | varchar(255) | 否 |  |  |
| certificate | varchar(255) | 否 |  |  |
| filetype | varchar(255) | 否 |  |  |
| starttime | datetime | 否 |  |  |
| endtime | datetime | 否 |  |  |

表3.5.2索引表1

| **键名** | **类型** | **唯一** | **紧凑** | **字段** | **基数** | **排序规则** | **空** | **注释** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | 是 | 否 | id | 2 | A | 否 |  |
| FK\_CERTIFICATE | BTREE | 否 | 否 | certificate | 2 | A | 否 |  |

表3.5.3certificateaudit表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **空** | **默认** | **注释** |
| id (主键) | int(11) | 否 |  |  |
| certificate | varchar(255) | 否 |  |  |
| uid | int(11) | 否 |  |  |
| cid | int(11) | 是 | *NULL* |  |
| filetype | varchar(255) | 否 |  |  |
| state | varchar(255) | 否 |  |  |

表3.5.4索引表2

| **键名** | **类型** | **唯一** | **紧凑** | **字段** | **基数** | **排序规则** | **空** | **注释** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | 是 | 否 | id | 3 | A | 否 |  |
| certificate | BTREE | 是 | 否 | certificate | 3 | A | 否 |  |
| FK\_CERTIFICATE\_UID | BTREE | 否 | 否 | uid | 3 | A | 否 |  |
| FK\_CERTIFICATE\_CID | BTREE | 否 | 否 | cid | 3 | A | 是 |  |

表3.5.5company表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **空** | **默认** | **注释** |
| id (主键) | int(11) | 否 |  |  |
| email | varchar(255) | 否 |  |  |
| company | varchar(255) | 否 |  |  |
| creditcode | varchar(255) | 否 |  |  |
| verifyinfo | varchar(255) | 否 |  |  |
| license | varchar(255) | 否 |  |  |
| state | varchar(255) | 否 |  |  |

表3.5.6索引表3

| **键名** | **类型** | **唯一** | **紧凑** | **字段** | **基数** | **排序规则** | **空** | **注释** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | 是 | 否 | id | 2 | A | 否 |  |
| company | BTREE | 是 | 否 | company | 2 | A | 否 |  |
| FK\_COMPANY\_EMAIL | BTREE | 否 | 否 | email | 2 | A | 否 |  |

表3.5.7 departmenttable表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **空** | **默认** | **注释** |
| id (主键) | int(11) | 否 |  |  |
| department | varchar(255) | 否 |  |  |

表3.5.8索引表4

| **键名** | **类型** | **唯一** | **紧凑** | **字段** | **基数** | **排序规则** | **空** | **注释** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | 是 | 否 | id | 1 | A | 否 |  |
| department | BTREE | 是 | 否 | department | 1 | A | 否 |  |

表3.5.9mapping表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **空** | **默认** | **注释** |
| fileid | varchar(255) | 否 |  |  |
| sealid | varchar(255) | 否 |  |  |

表3.5.10索引表5

| **键名** | **类型** | **唯一** | **紧凑** | **字段** | **基数** | **排序规则** | **空** | **注释** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| fileid | BTREE | 否 | 否 | fileid | 12 | A | 否 |  |
| sealid | BTREE | 否 | 否 | sealid | 12 | A | 否 |  |

表3.5.11sealedfile表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **空** | **默认** | **注释** |
| id (主键) | int(11) | 否 |  |  |
| label | varchar(255) | 否 |  |  |
| encodefile | varchar(2049) | 否 |  |  |
| uid | int(11) | 否 |  |  |
| cid | int(11) | 是 | *NULL* |  |

表3.5.12索引表6

| **键名** | **类型** | **唯一** | **紧凑** | **字段** | **基数** | **排序规则** | **空** | **注释** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | 是 | 否 | id | 1 | A | 否 |  |
| idx\_sealedfile\_label | BTREE | 是 | 否 | label | 1 | A | 否 |  |
| FK\_SF\_UID | BTREE | 否 | 否 | uid | 1 | A | 否 |  |
| FK\_SF\_CID | BTREE | 否 | 否 | cid | 1 | A | 是 |  |

表3.5.13sealtable表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **空** | **默认** | **注释** |
| id (主键) | int(11) | 否 |  |  |
| sealid | varchar(255) | 否 |  |  |
| sealname | varchar(255) | 否 |  |  |
| uid | int(11) | 否 |  |  |
| certificateid | varchar(255) | 否 |  |  |
| sealtype | varchar(255) | 否 |  |  |
| info | varchar(255) | 否 |  |  |

表3.5.14索引表7

| **键名** | **类型** | **唯一** | **紧凑** | **字段** | **基数** | **排序规则** | **空** | **注释** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | 是 | 否 | id | 16 | A | 否 |  |
| sealid | BTREE | 是 | 否 | sealid | 16 | A | 否 |  |
| sealname | BTREE | 是 | 否 | sealname | 16 | A | 否 |  |
| FK\_SEAL\_UID | BTREE | 否 | 否 | uid | 16 | A | 否 |  |
| FK\_SEAL\_CERTIFICATEID | BTREE | 否 | 否 | certificateid | 16 | A | 否 |  |

表3.5.15users表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段** | **类型** | **空** | **默认** | **注释** |
| id (主键) | int(11) | 否 |  |  |
| username | varchar(255) | 否 |  |  |
| email | varchar(255) | 否 |  |  |
| passwd | varchar(255) | 否 |  |  |
| cellphone | varchar(255) | 否 |  |  |
| department | varchar(255) | 否 |  |  |
| power | varchar(255) | 否 |  |  |
| isverify | varchar(255) | 否 |  |  |
| state | varchar(255) | 否 |  |  |

表3.5.16索引表8

| **键名** | **类型** | **唯一** | **紧凑** | **字段** | **基数** | **排序规则** | **空** | **注释** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| PRIMARY | BTREE | 是 | 否 | id | 5 | A | 否 |  |
| email | BTREE | 是 | 否 | email | 5 | A | 否 |  |
| FK\_USER\_DEPARTMENT | BTREE | 否 | 否 | department | 5 | A | 否 |  |

### 3.5.2数据库E-R图

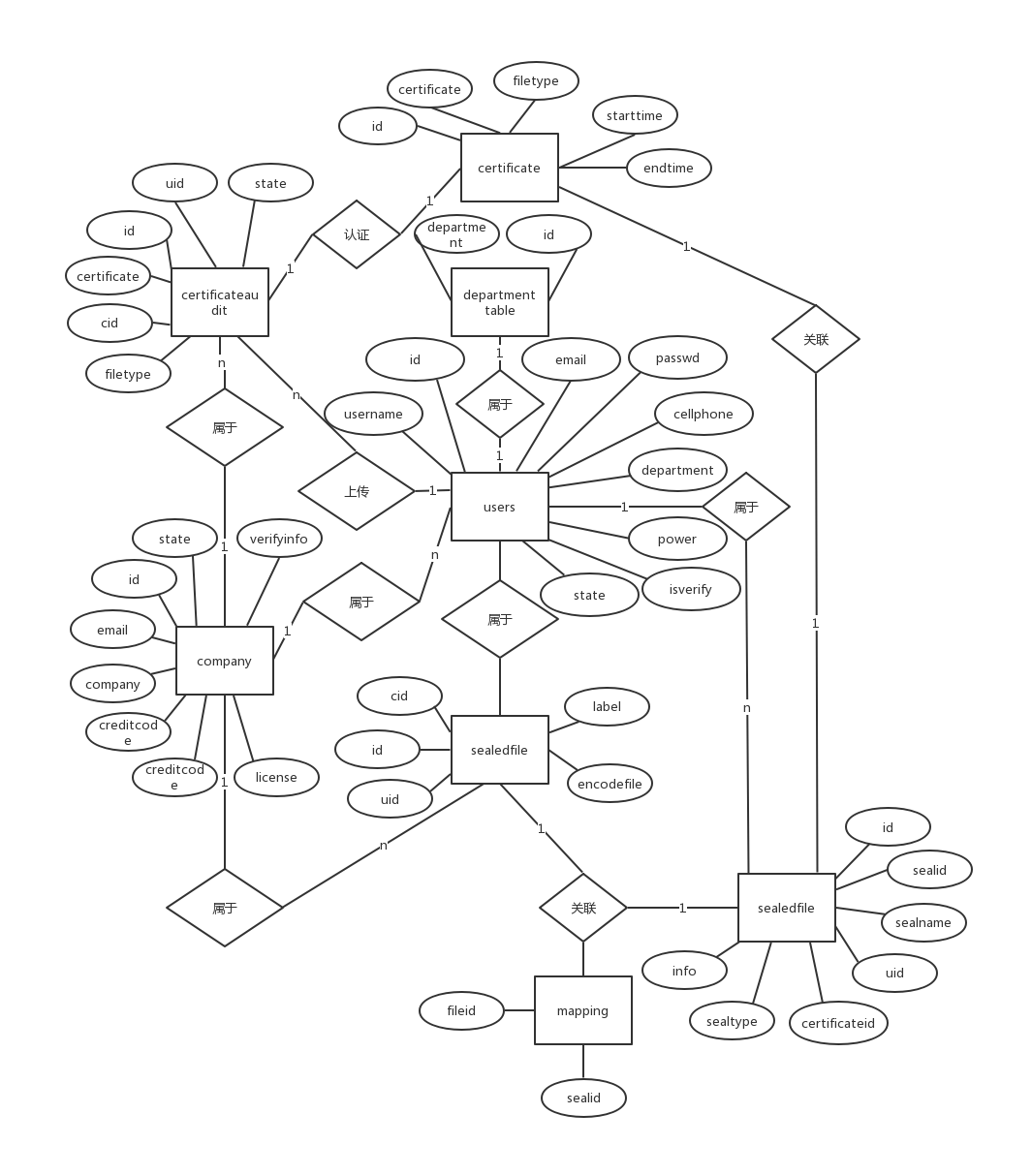


图3.5数据库E-R图

# 详细设计



图4-1主界面截图

此图为本系统软件主界面原型图。其中菜单可以使用本系统的所有命令，用户可以通过这些命令对基于Linux的面向PDF文档的电子印章系统进行交互。菜单中包括很多隐藏命令。菜单主要命令按功能分为一下几类，PDF文档操作命令、电子印章操作命令、用户帮助文档。

工具栏上为用户常用命令。

文件预览为PDF文档的预览，用户可以通过文件预览来选择跳到某一页。是PDF文档的提前预览。

页面显示用于显示某一页面，且每次只能显示一页，同时支持垂直滚动。以及调整页面大小。

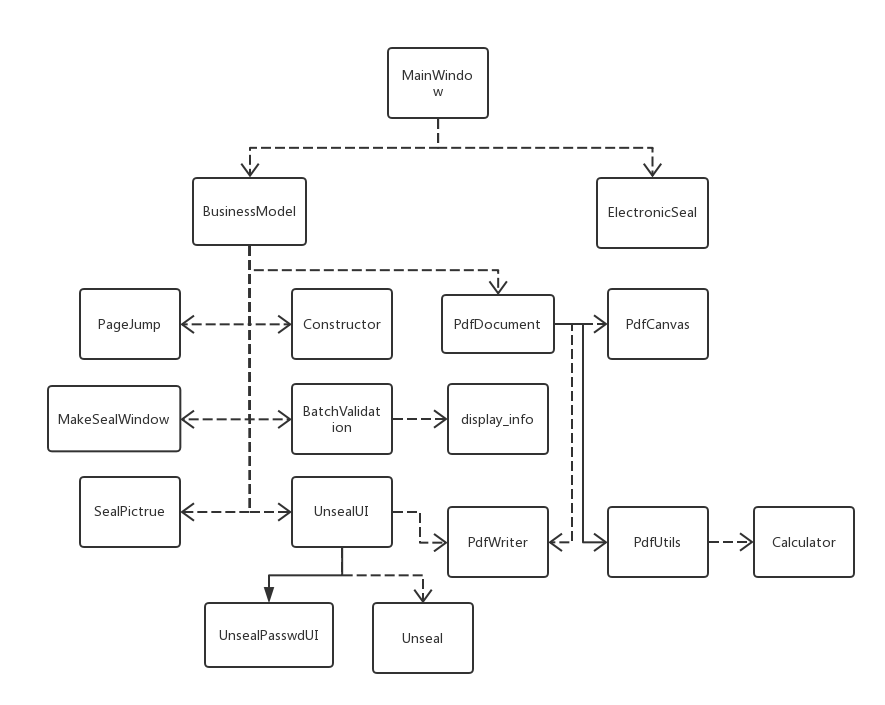


图4-2总体类图

图4-3用户活动图

## 4.1 PDF文档操作模块设计

此类命令用于对PDF文档进行基本操作，主要包括打开文件、关闭文件、对文件页面进行调整、定位页码、保存文档。

### 4.1.1打开PDF文档

要打开PDF文档，首先需要从MainWindow中调用BusinessModel类的openFile方法，传入相应的参数；openFile方法会new一个PDFDocument类，然后调用PDFDocument类的openPDF方法；在openPDF中会调用PdfUtils类的getPdfDocument方法，通过开源库poppler的Poppler::Document::load方法、page方法，Poppler::Page的renderToImage方法，实现对PDF文档内容的提取并转换成图片，由于PdfUtils继承了QThread类，在加载了文档后，用另一线程加载目录，避免主程序的停滞，显示页面后，openPDF方法还调用了PdfUtils类的countInfoAbstract方法和loadSealInfo方法，用于获取PDF文件的信息摘要、印章操作增量更新以及证书信息并缓存起来，最后通过调用PDFCanvas类的drawCanvas方法实现对PDF文档的绘制，到此打开文档完毕。

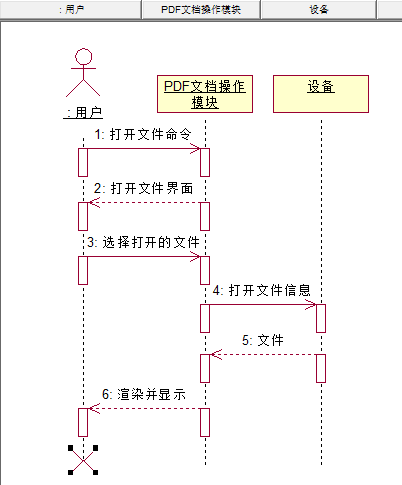
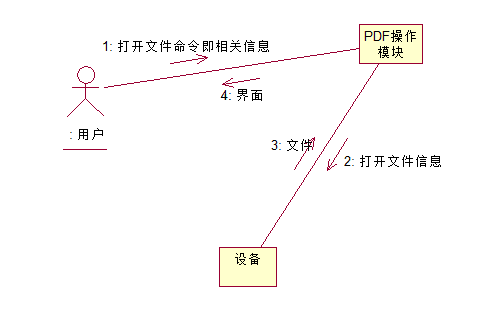


图4.1.1-1打开文件时序图 图4.1.1-2打开文件协作图

### 4.1.2页面大小调整

要实现页面大小的调整，首先要为放大比例设置一个初值，初值作为页面的原始大小的缩放比例，然后设置增加量和减少量，用于实现当点击特定增大或缩小按钮时，能按比例缩放，同时设置一个值，使文档图片能放大至铺满整个显示区域。根据放大比率的变化，调用PDFCanvas类的zoomCanvas方法实现对页面大小缩放功能的实现。

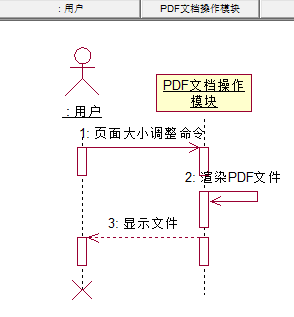
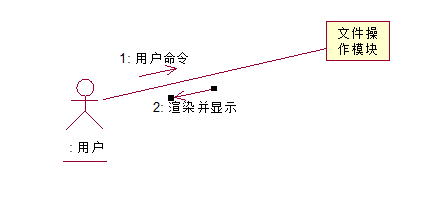


图4.1.2-1页面大小调整时序图 图4.1.2-2页面大小调整协作图

### 4.1.3页面跳转

页面跳转功能是通过调用BusinessModel类的jumpPage方法传入相应参数，然后在jumpPage中调用PDFDocument类的jumpPage方法，jumpPage方法会检测将要跳转的页面是否在图片缓存中。如果在就直接调用pdfCanvas对象对文档显示区域进行渲染；如果不在就调用PdfUtils类的getPdfDocument方法和getPdfImage方法装载页面图片，然后再在文档显示区域显示。

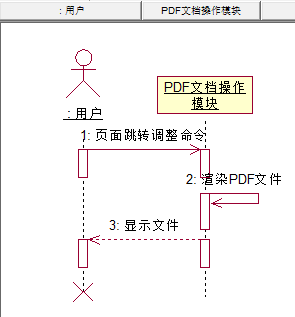


图4.1.3页面跳转时序图

### 4.1.4保存文档

文档保存和另存为功能是通过调用BusinessModel类的saveFile和saveAs方法，两者都会调用PDFDocument类的savePDF方法，差别在于保存只需传入当前文件名作为参数，另存为则需要再传入文件保存的目标地址。savePDF方法会调用PdfUtils类的openPdf和saveOutput方法，通过Poppler库的Document类中的PDFConverter方法获取PDF文档的备份，用setPDFOptions方法设置备份保存修改，在用setOutPutFileName方法设置要输出的文件名（文件地址），通过convert方法输出到硬盘中，即可实现保存。

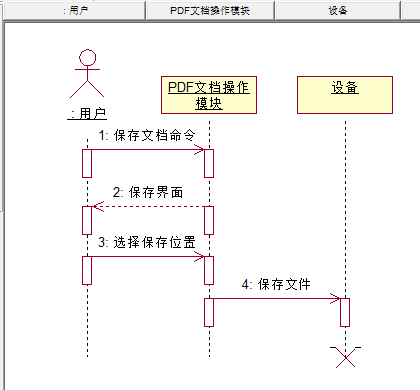
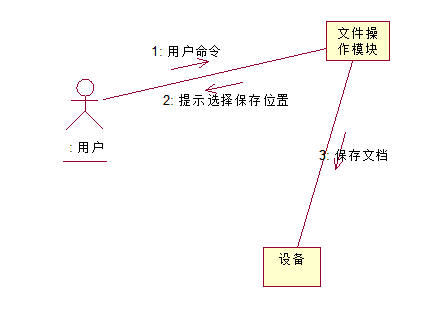


图4.1.4-1保存文档时序图 图4.1.4-2保存文档协作图

## 4.2电子印章操作

此类命令用来对电子印章进行操作，主要包括制作电子印章、盖章、撤章、验证等功能。

### 4.2.1制作电子印章

系统中制章功能是通过制章制章响应函数实现，界面显示模块接收用户制章命令。随后转到制章模块，并提示用户输入相关信息来制作电子印章，如：电子印章图片所在位置、证书所在位置、证书密钥位置、用户基本信息、电子印章密码等。在输入证书位置后，会向服务器发送请求，检查证书是否已经过认证，如果未认证则无法继续进行，同时客户端会调用openssl校验证书是否为非有效CA颁布的证书。导入证书私钥时，会从私钥中导出公钥。制章模块将用户输入的信息以及证书校验信息和证书公钥信息联合起来构造电子印章，并放入用户选择的位置。

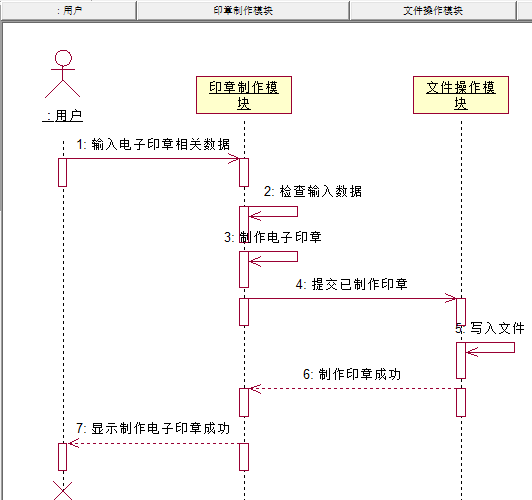
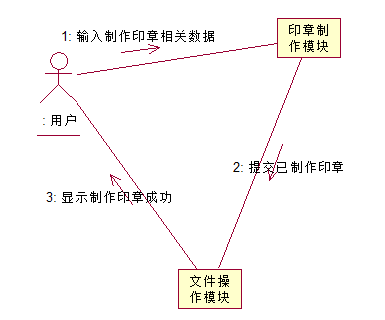


图4.2.1-1制作电子印章时序图 图4.2.1-2制作电子印章协作图

### 4.2.2盖章

系统中的盖章功能是通过盖章响应函数实现，界面显示模块接收用户盖章命令，随后界面显示模块将盖章相关信息（盖章页面以及用户选择盖章的位置坐标）传给盖章模块。盖章模块暂时缓存下相关信息以便后边使用，随后盖章模块从设备中取出电子印章数据。此过程中需要用户选择具体设备下存储的电子印章并且输入电子印章密码。只有用户密码争取却才能继续执行，否则推出盖章模块。盖章模块根据电子印章数据以及盖章相关信息组织电子签章数据，并将电子签章写入PDF文档中，同时通知PDF文件渲染模块页面已更新，PDF文件渲染模块对PDF文档进行重新渲染并将渲染好的文件显示给用户。盖章文件信息和所用证书信息将会被发送至服务器端。

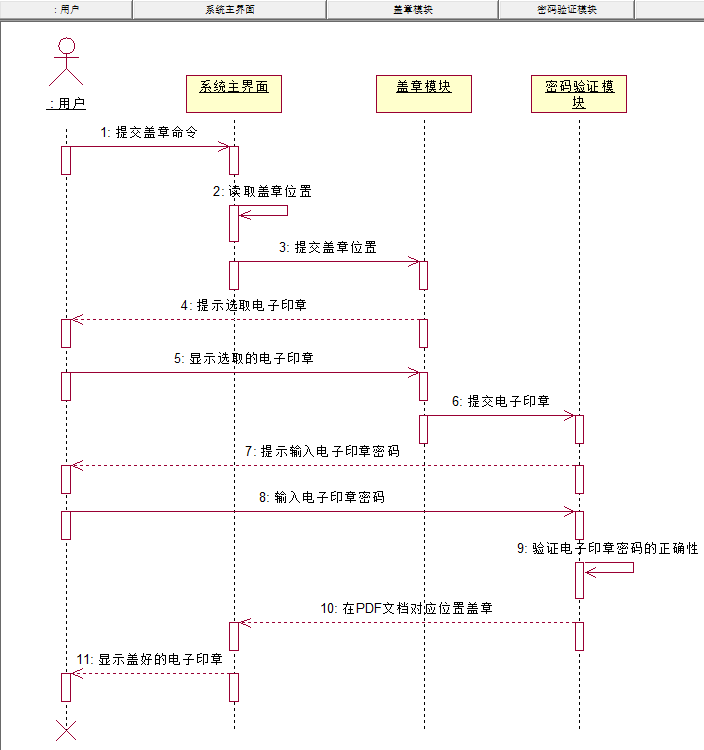


图4.2.2-1盖章时序图

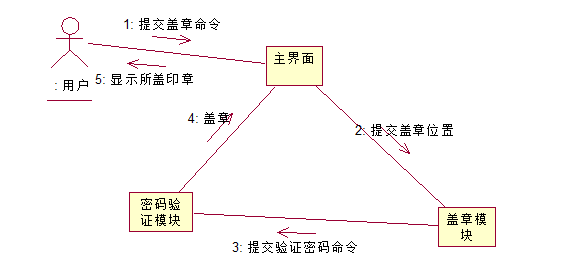


图4.2.2-2盖章协作图

### 4.2.3撤章

系统中的撤章功能是通过撤章响应函数实现，界面显示模块接收用户撤章命令，并将PDF文档中的电子签章信息组织好传递给撤章模块。撤章模块收到电子签章信息后显示出来当用户选择具体的电子签章时撤章模块将签章渲染好并展示给用户查看，同时要求输入该电子签章对应的电子印章的密码来验证用户信息。验证成功撤章模块将PDF文档中的电子签章相关对象进行修改。并通知界面显示模块更新PDF文档。界面显示模块重新渲染PDF文件后展示给用户查看。不过撤章有一个前提条件，即服务器端不能有该文件的盖章信息，如果有，就不允许撤章，保证文件的完整性。

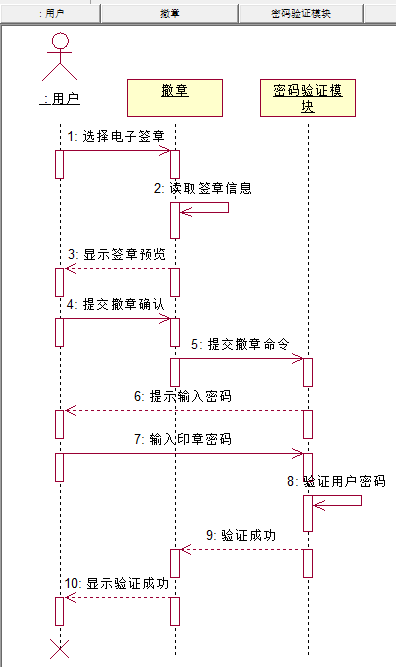
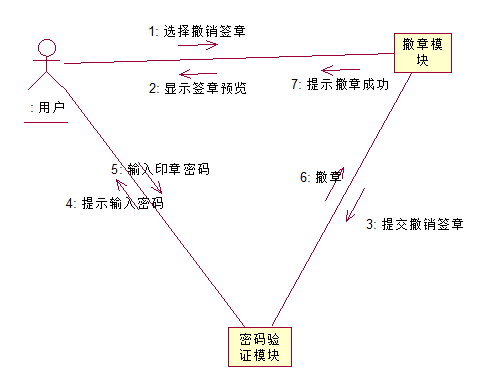


图4.2.3-1撤章时序图 图4.2.3-2撤章协作图

### 4.2.4验证

系统中的验证功能是通过验证响应函数实现，界面显示模块接收用户验证命令，并组织验证相关信息并连同验证命令一起传递给验证模块。验证模块收到界面显示模块传递的传输后先进行初步验证并显示给用户，当用户选择具体的电子印章后验证模块对用户选择的具体电子印章进行验证并显示给用户查看。当用户选择查看证书时验证模块组织CA证书相关信息并传递给证书验证模块，证书验证模块接收到信息后组织证书结构并显示给用户查看，当用户选择要查看的证书后对证书进行验证并显示给用户查看。同时，在打开验证界面时，会向服务器端发送文件的信息摘要至服务器，获取文件的盖章信息，用户可通过服务器端回传信息按钮查看服务器端传回的盖章文件中的印章信息。

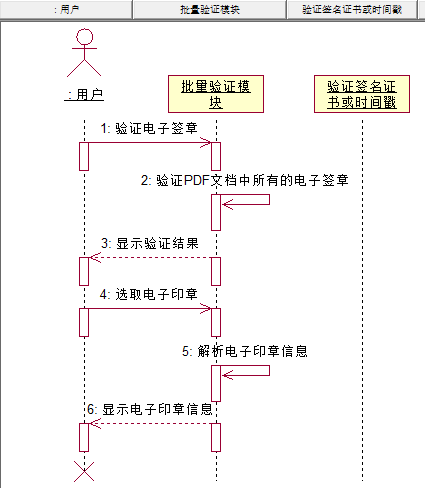
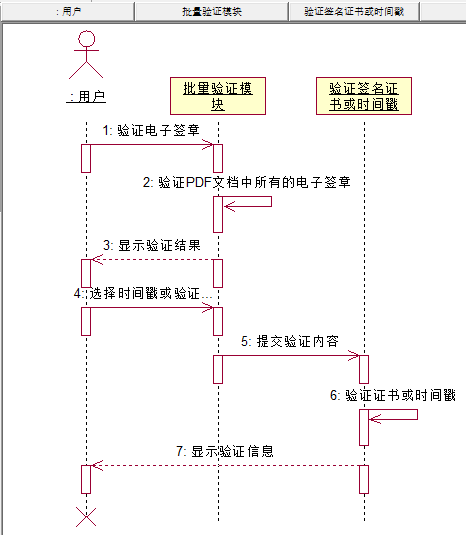


图4.2.4-1验证时序图 图4.2.4-2用户签名证书时序图

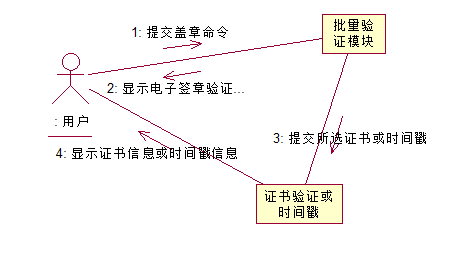
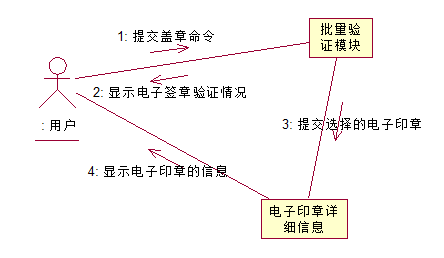


图4.2.4-3验证协作图 图4.2.4-4用户签名证书协作图

## 客户端网络相关操作

### 4.3.1登陆

登录功能通过Http协议实现，在用户输入账号密码后，通过Http协议向服务器发送账号密码数据，服务器接收后进行账号密码的校验，并返回登陆结果，如果登陆成功就跳转到主界面，并在界面左下角显示用户的昵称。

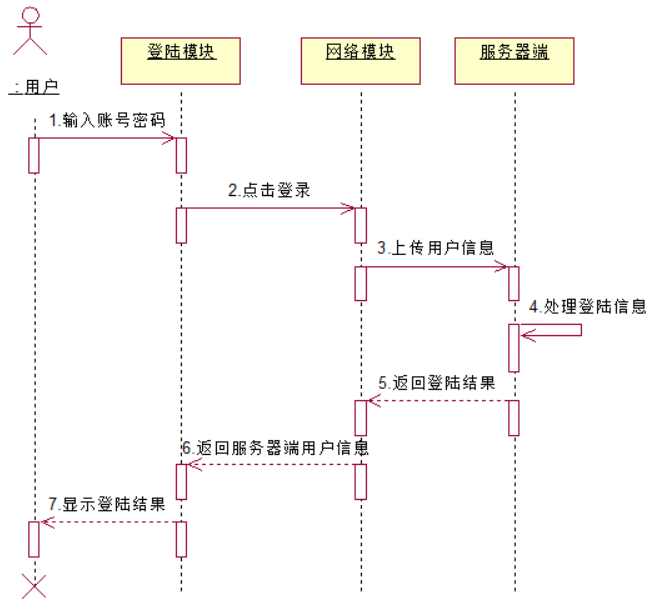


图4.3.1-1登陆时序图

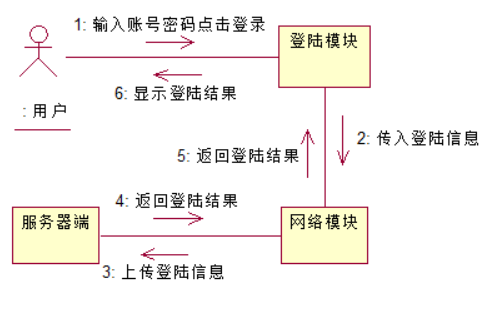


图4.3.1-2登陆协作图

### 4.3.2网络传输

该功能用于点击注册按钮时跳转至官方主页，发送用户登录信息，发送进行电子印章相关操作时的产生的相关数据，包括印章的信息，盖章文件的信息，和获取用户登陆结果，获取印章制作结果，获取盖章文件的验证结果。

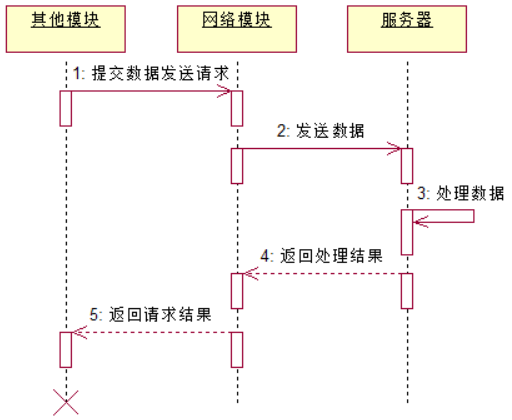


图4.3.2 网络传输时序图

## 网页端具体设计

### 4.4.1用户公司信息及证书信息上传

用户点击上传按钮后进行文件的选择，当文件上传完后将文件相关信息以及用户信息以前传递给后台服务器，服务器读取到相关信息后将将文件保存到本地静态资源文件夹里，然后把所有信息整合存入数据库。

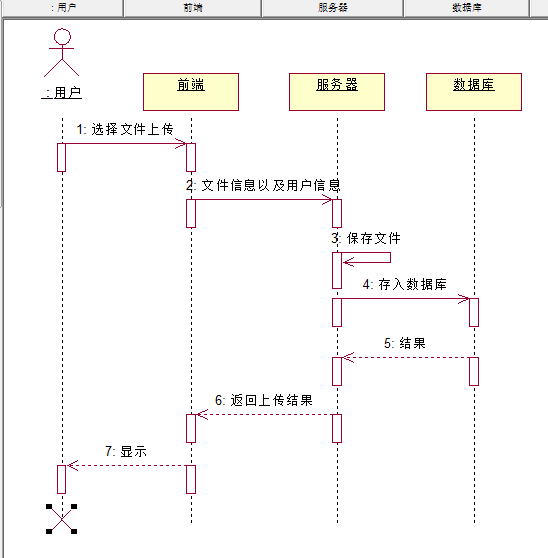
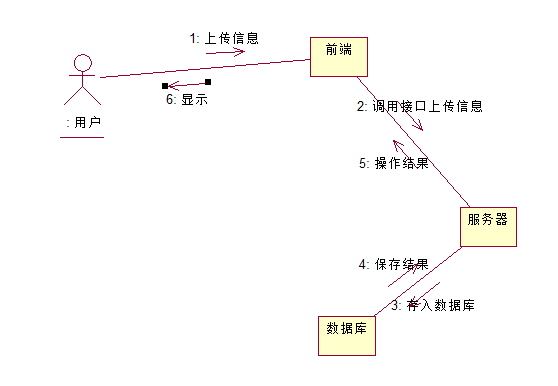


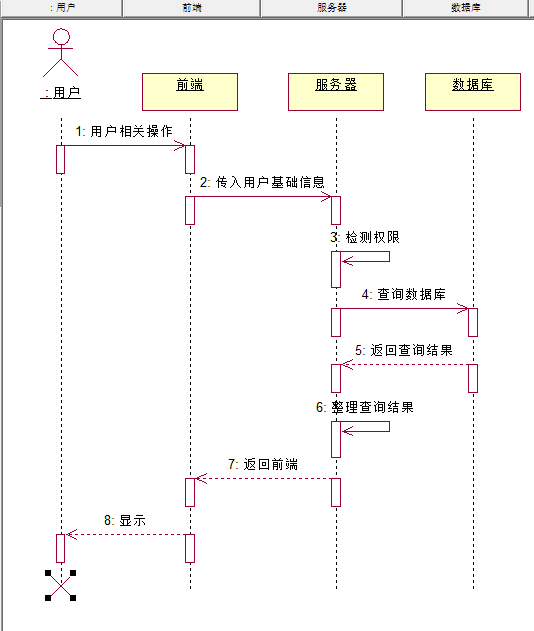
图4.4.1-1公司信息及证书信息上传时序图



4.4.1-2用户公司及证书信息上传协作图

### 4.4.2管理员用户信息管理

管理员点击用户信息查看空间后前端想后端发出指令，服务器响应后检测用户权限是否能够操作此功能给，如果可以则通过数据库查询到信息存入JSON中返回给前端进行显示。



4.4.2-1管理员用户信息管理时序图

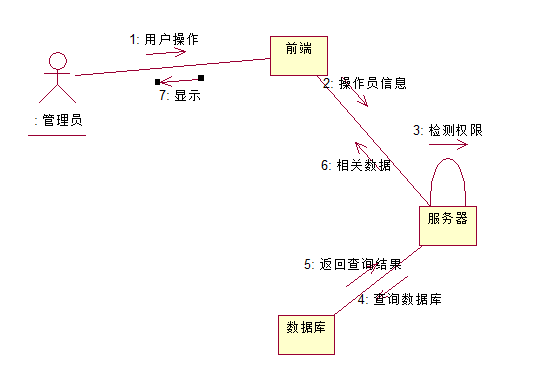


图4.4.2-2管理员用户信息管理协作图

### 4.4.3管理员审核公司信息

将所有待审核的公司信息通过表格的方式显示出来，管理员可自行选择其中一个信息点击详情进行审核，点击详情后会把所有的审核信息显示出来给管理员查看。然后管理员根据信息的详细情况进行认可或者驳回。

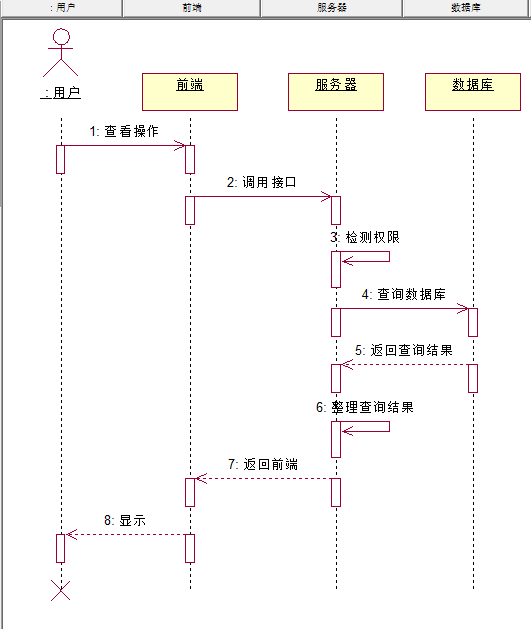
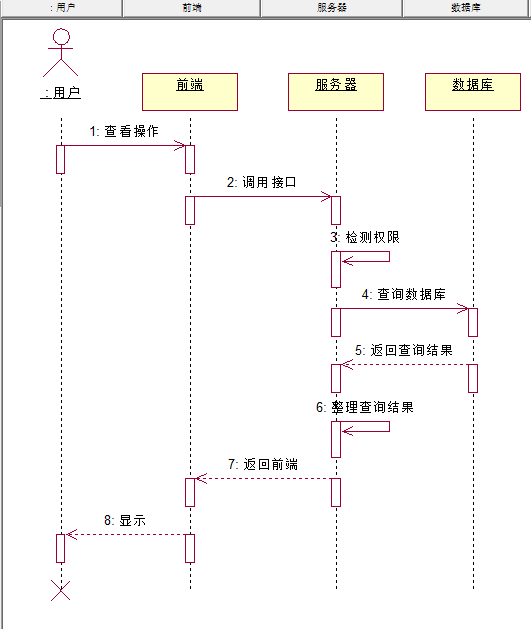


图4.4.3-1管理员审核公司信息

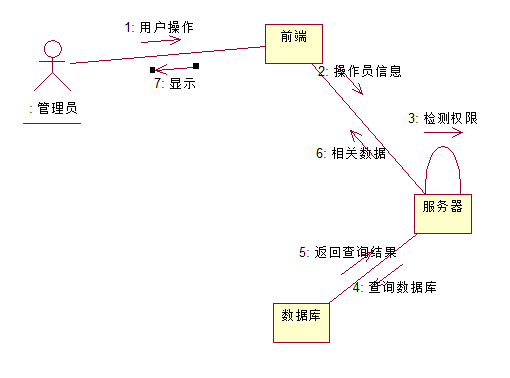
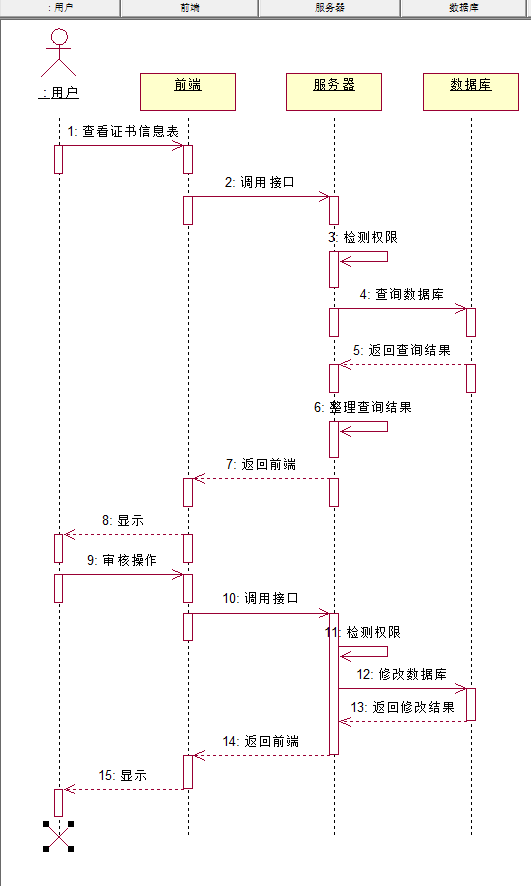


图4.4.3-2管理员审核公司信息协作图

### 4.4.4管理员审核证书信息

将所有待审核的证书信息通过表格的方式显示出来，管理员可自行选择其中一个信息下载到本地进行审核。然后管理员根据信息的详细情况进行认可或者驳回。



4.4.4-1审核证书信息时序图

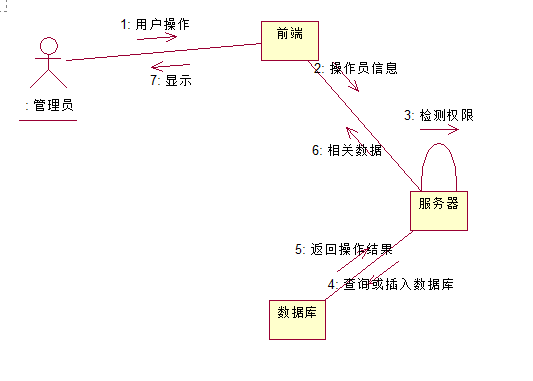


图4.4.4-2管理员审核证书信息协作图

# 系统实现

基于Linux的面向PDF文档的电子印章系统客户端要求用户的最低配置为：

Qt5.9.5及以上、安装openssl、安装libpoppler-dev。服务器端要求安装Java。

## 5.1 PDF文档操作模块

### 5.1.1思路

基于开源库Poppler实现对PDF文档的加载和提取图片映射，使得pdf文档能与图片的形式显示。在打开文档的同时，新建一个线程用于文件目录的加载。将已读的页面进行缓存，可减少对重复页面的加载，在进行页面跳转的时候首先检查是否有页面图片已缓存，若已有缓存则从中进行加载显示，若没有则调用Poppler库的类和方法进行加载。页面的缩放功能则是通过设置图片大小重新在显示界面进行显示。文档的保存是通过Poppler库进行文档备份的创建，然后将备份输出到指定文件目录下。

### 5.1.2相关类及其描述

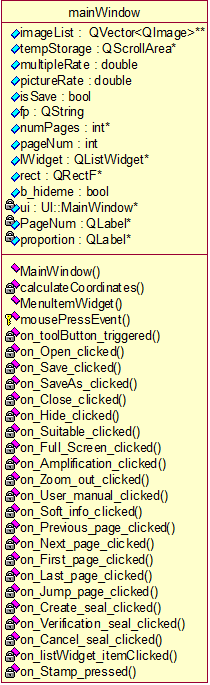
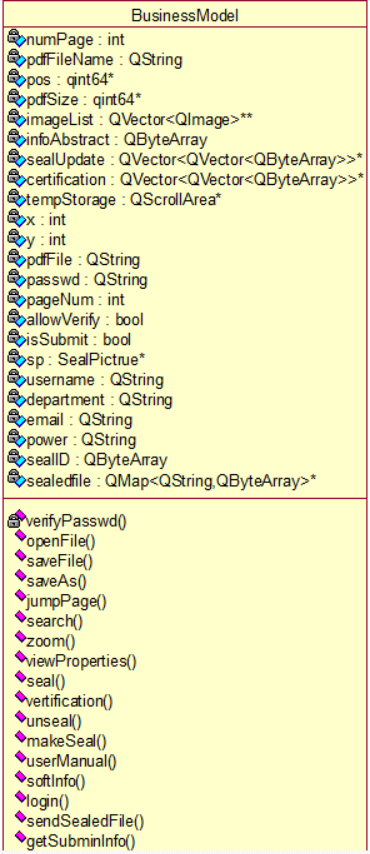
 

图5.1.2-1主窗口类图 图5.1.2-2控制终端类类图

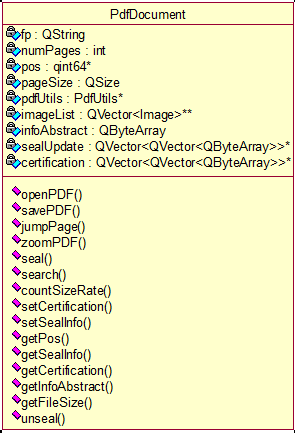
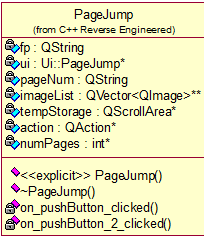
 

图5.1.2-3 PDF文档处理类类图 图5.1.2-4页面跳转类类图

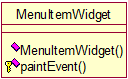


图5.1.2-5文件按钮列表类类图

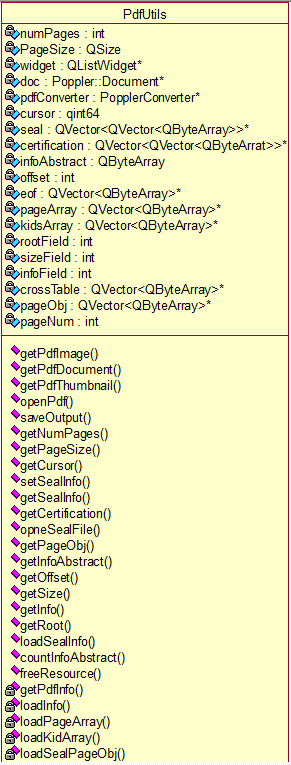


图5.1.2-6 PDF工具类类图

由于构造方法中基本是对属性进行赋值，所以除了主界面和文件按钮列表类的构造方法，其他类的构造方法没在各类的方法表中列出并进行描述。

表5.1.2-1主窗口类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| imageList | 用于临时存储转换得到的pdf图片 |
| tempStorage | 用于显示文档的widget |
| multipleRate | 放大缩小的倍率 |
| pictureRate | 实际pdf大小与图片大小的比例 |
| isSave | 用于判断当前文档是否被保存 |
| fp | 用于临时存储pdf文件地址 |
| numPages | 用于存储当前PDF文件的总页数 |
| pageNum | 当前显示的PDF文档的页面的页码 |
| lWidget | 缩略图 |
| rect | pdf坐标 |
| b\_hideme | 缩略图列表的是否被隐藏的判断标记 |
| ui | 主界面的实例化对象 |
| PageNum | 用于显示PDF文件的页码 |
| Proportion | 用于显示缩略图的当前的缩放比例 |

表5.1.2-2主窗口类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| calculateCoordinates | QPoint, QVector<int>\* | void | 计算鼠标点击位置在界面上的实际坐标 |
| MainWindow | QWidget \* | explicit | 类的构造方法，其中设置了各个组件的样式，对界面进行了美化。 |
| MenuItemWidget | const QPixmap&，  const QString&，  QWidget \* | void | 向点击文件按钮后列出来的菜单列表添加列表项 |
| mousePressEvent | QMouseEvent\* | void | 鼠标点击事件 |
| on\_toolButton\_triggered | QAction \* | void | 隐藏目录按键实现 |
| on\_Open\_clicked | 无 | void | 打开文件 |
| on\_Save\_clicked | 无 | void | 保存文件 |
| on\_SaveAs\_clicked | 无 | void | 文件另存为 |
| on\_Close\_clicked | 无 | void | 关闭 |
| on\_Hide\_clicked | 无 | void | 隐藏目录 |
| on\_Suitable\_clicked | 无 | void | 适合页面 |
| on\_Full\_Screen\_clicked | 无 | void | 全屏 |
| on\_Amplification\_clicked | 无 | void | 放大 |
| on\_Zoom\_out\_clicked | 无 | void | 缩小 |
| on\_User\_manual\_clicked | 无 | void | 用户手册 |
| on\_Soft\_info\_clicked | 无 | void | 软件信息 |
| on\_Previous\_page\_clicked | 无 | void | 上一页 |
| on\_Next\_page\_clicked | 无 | void | 下一页 |
| on\_First\_page\_clicked | 无 | void | 跳转至第一页 |
| on\_Last\_page\_clicked | 无 | void | 跳转至最后一页 |
| on\_Jump\_page\_clicked | 无 | void | 跳转页面 |
| on\_Create\_seal\_clicked | 无 | void | 制作电子印章 |
| on\_Verification\_seal\_clicked | 无 | void | 验证电子印章 |
| on\_Cancel\_seal\_clicked | 无 | void | 撤销电子印章 |
| on\_listWidget\_itemClicked | QListWidgetItem \* | void | 缩略图目录点击事件，点击相应缩略图跳转至相应界面 |
| on\_Stamp\_pressed | 无 | void | 盖章 |

表5.1.2-3控制终端类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| numPages | 当前文档总页数 |
| pdfSize | 页面大小 |
| pos | 印章坐标 |
| pdfFileName | PDF文档名 |
| infoAbstract | 文件信息摘要 |
| sealUpdate | pdf中已存在印章操作更新 |
| certification | 印章的证书信息 |
| imageList | 一缓存页面图片 |
| x | 盖章位置x轴坐标 |
| y | 盖章位置y轴坐标 |
| tempStorage | 文件图片显示区 |
| allowVerify | 是否允许验证 |
| isSubmit | 盖章信息是否提交 |
| sp | 当前使用的印章 |
| username | 用户昵称 |
| department | 用户所属部门 |
| email | 用户邮箱 |
| power | 用户权限 |
| sealID | 印章编号 |
| sealedfile | 盖章后的文件信息 |

表5.1.2-4控制终端类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| openFile | QString，  QVector<QImage>\*\*，  QScrollArea，  Int，  QListWidget\*，  Double | void | 调用文档打开功能 |
| saveFile | QString | void | 调用保存当前文档功能 |
| saveAs | QString，  QString | void | 调用当前文档另存为功能 |
| jumpPage | Int，QVector<QImage>\*\* ，  QScrollArea\*，  QString， | void | 调用页面跳转功能 |
| zoom | QVector<QImage>\*\*，  int，  double，  QScrollArea\* | void | 调用页面缩放功能 |
| seal | QVector<QImage>\*\*，  QScrollArea\*，  Int，  Int，  QString，  QString，  QString，  Int， | void | 调用盖章功能 |
| vertification | 无 | void | 调用文档印章验证功能 |
| Unseal | QVector<QImage>\*\*，  QScrollArea\*，  int | void | 调用撤章功能 |
| makeSeal | 无 | void | 调用印章制作功能 |
| userManual | 无 | void | 调用用户说明书显示功能 |
| softInfo | 无 | void | 调用软件介绍显示功能 |
| freeResource | 无 | void | 调用临时资源释放功能 |
| verifyPasswd | QString，  QByteArray | bool | 验证输入的印章盖章密码 |
| login | 无 | void | 用户登录 |
| sendSealedFile | 无 | 无 | 发送已盖章文件信息 |

表5.1.2-5 PDF文档处理类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| fp | PDF文档文件名 |
| numPages | 文档总页数 |
| pos | 印章盖章坐标 |
| pageSize | PDF文档页面图片大小 |
| pdfUtils | PDF工具类实例化的对象 |
| imageList | 已缓存的PDF页面图片的列表 |
| infoAbstract | 文件信息摘要 |
| sealUpdate | pdf中已存在印章操作增量更新 |
| certification | 印章的证书信息 |

表5.1.2-6 PDF文档处理类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| openPDF | QVector<QImage>\*\*  QScrollArea\*  QListWidget\*  double | int | 打开PDF文档并返回pdf文档的总页数 |
| savePDF | QString，  QString | void | 保存文档，若传进来的是两个参数就是执行另存为 |
| jumpPage | Int，  QVector<QImage>\*\*，  QScrollArea\* | void | 页面跳转 |
| zoomPDF | QVector<QImage>\*\*，  QScrollArea\*，  Int，  double | void | 缩放pdf页面图片 |
| seal | QVector<QImage>\*\*  QScrollArea\*  int  int  SealPictrue\*  QString  int | void | 盖章 |
| countSizeRate | 无 | double | 计算pdf图片实际大小和渲染后的大小的比值 |
| setCertification | QVector<QVector<QByteArray>>\* | void | 加载证书信息 |
| setSealInfo | QVector<QVector<QByteArray>>\* | void | 加载印章信息 |
| getPos | 无 | qint64 | 获取第一次印章增量更新前的文件光标位置 |
| getSealInfo | 无 | QVector<QVector<QByteArray>>\* | 获取已存在印章的信息 |
| getCertificatio | 无 | QVector<QVector<QByteArray>>\* | 获取印章证书信息 |
| getInfoAbstract | 无 | QByteArray | 获取文件信息摘要 |
| getFileSize | 无 | qint64 | 获取文件大小 |
| unseal | QVector<QImage>\*\*  QScrollArea\*  int  QString | void | 撤章 |

表5.1.2-7页面跳转类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| fp | PDF文档文件名 |
| ui | 页面跳转UI |
| pageNum | 传入的页码 |
| imageList | 已缓存PDF文档页面图片列表 |
| tempStorage | 显示PDF文档的widget |
| action | 页码显示区块 |
| numPages | 总页数 |

表5.1.2-8页面跳转类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| on\_pushButton\_clicked | 无 | void | 确认按钮事件 |
| on\_pushButton\_2\_clicked | 无 | void | 取消按钮事件 |

表5.1.2-9文件按钮列表类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| MenuItemWidget | const QPixmap&，  const QString&，  QWidget \* | void | 类的构造方法，同时设置文件按钮点击时列出的按钮列表的样式。 |
| paintEvent | QPaintEvent\* | Void | 设置文件菜单列表的背景色 |

表5.1.2-10 PDF工具类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| numPages | PDF文档总页数 |
| pageSize | 页面实际大小 |
| widget | 加载缩略图目录 |
| doc | Poppler::Document类实例化的对象 |
| pdfConverter | Poppler::PDFConverter类实例化的对象 |
| cursor | 印章增量更新开始前的光标位置 |
| seal | pdf中已存在印章操作增量更新 |
| certification | 印章的证书信息 |
| infoAbstract | 文件信息摘要 |
| offset | PDF文件原文件偏移量 |
| eof | PDF文件文件尾 |
| pagesArray | PDF文件页面字段对象(原始文件目录) |
| kidsArray | 目录 |
| rootField | PDF文件Root字段 |
| sizeField | PDF文件Size字段 |
| nfoField | PDF文件Info字段 |
| crossTable | PDF文件交叉引用表 |
| pageObj | PDF文件页面对象 |
| pageNum | 盖章页面的页数 |

表5.1.2-11 PDF工具类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| getPdfImage | Int，  QVector<QImage>\*\* | void | 将pdf的文档内容转换成图片 |
| getPdfDocument | QString | void | 获取pdf文档及其页面大小的信息和页面总数 |
| getPdfThumbnail | Int | QImage | 获取PDF文档图片的缩略图 |
| openPdf | QString | void | 打开pdf文档，用于另存为 |
| saveOutput | QString | void | 另存为 |
| getNumPages | 无 | int | 获取pdf总页码 |
| getPageSize | 无 | QSize | 获取pdf页面大小 |
| getCursor | 无 | qint64 | 获取印章增量更新前的光标位置 |
| setSealInfo | QVector<QVector<QByteArray>>\* | void | 装载以保存的印章信息 |
| getSealInfo | 无 | QVector<QVector<QByteArray>>\* | 获取已存在印章的信息 |
| getCertification | 无 | QVector<QVector<QByteArray>>\* | 获取印章证书信息 |
| openSealFile | QString | bool | 打开要盖章的文件 |
| getPageObj | 无 | QVector<QByteArray>\* | 获取PDf间接页面对象 |
| getInfoAbstract | 无 | QByteArray | 获取文件信息摘要 |
| getOffset | 无 | int | 获取文件偏移量 |
| getSize | 无 | int | 获取文件间接对象总数 |
| getInfo | 无 | int | 获取文件信息间接对象号 |
| getRoot | 无 | int | 获取文件的Root字段 |
| loadSealInfo | QString | void | 获取印章信息和证书信息 |
| countInfoAbstract | QString | void | 计算信息摘要 |
| freeResource | 无 | bool | 释放资源 |
| run | 无 | void | 该线程用于加载缩略图 |
| getPdfInfo1 | QString | void | 获取盖章pdf信息的方法（还没盖过章的） |
| getPdfInfo2 | QString | void | 获取盖章pdf信息的方法（盖过一次章以上的） |
| loadInfo | 无 | bool | 加载所需信息 |
| loadPagesArray | QString | bool | 加载文件页面的目录 |
| loadKidArray | QString | bool | 加载各页面的kid(即各页面间接对象的编号) |
| loadSealPageObj | QString | bool | 加载盖章页面的间接对象 |

### 5.1.3核心代码

#### 5.1.3.1加载PDF文档

void PdfUtils::getPdfDocument(QString filepath)

{

numPages = 0;

doc = Poppler::Document::load(filepath);

// 更加清晰

doc->setRenderHint(Poppler::Document::Antialiasing);

doc->setRenderHint(Poppler::Document::TextAntialiasing);

numPages = doc->numPages();

Poppler::Page\* pdfPage = doc->page(0);

pageSize = pdfPage->pageSize();

delete pdfPage;

}

#### 5.1.3.2将PDF文档转为图片

void PdfUtils::getPdfImage(int pageNumber,QVector<QImage>\*\* imageList)

{

QImage image;

Poppler::Page\* pdfPage;

pdfPage = doc->page(pageNumber);

image = pdfPage->renderToImage(128,128);

(\*imageList)->replace(pageNumber, image);

delete pdfPage;

}

#### 5.1.3.3获取PDF文档缩略图

QImage PdfUtils::getPdfThumbnail(int pageNumber)

{

QImage image;

Poppler::Page\* pdfPage;

pdfPage = doc->page(pageNumber);

image = pdfPage->renderToImage(30,30);

delete pdfPage;

return image;

}

#### 5.1.3.4创建新的线程加载目录

void PdfUtils::run()

{

for(int i = 0; i < getNumPages(); i++)

{

if(widget == nullptr)

{

return;

}

QPixmap\* pixmap=new QPixmap();

pixmap->convertFromImage(getPdfThumbnail(i));

QListWidgetItem \*item = new QListWidgetItem;

item->setIcon(QIcon(\*pixmap));

item->setText(QString::number(i+1,10));

widget->addItem(item);

}

}

#### 5.1.3.5创建PDF文档备份

void PdfUtils::openPdf(QString filePath)

{

this->doc = Poppler::Document::load(filePath);

this->pdfConverter = this->doc->pdfConverter();

this->pdfConverter->setPDFOptions(Poppler::PDFConverter::WithChanges);

}

#### 5.1.3.6输出文档备份

void PdfUtils::saveOutput(QString filePath)

{

this->pdfConverter->setOutputFileName(filePath);

this->pdfConverter->convert();

}

## 5.2 电子印章操作模块

### 5.2.1思路

印章的制作采用的是JPG和JPEG图片，制作方法是将用户在制作印章时填写的信息、制作印章所使用的证书，和从公钥中提取出的私钥一并写入到JPG或JPEG图片的末尾。在制作过程中，会使用openssl检查证书是否是从有效CA处申请的证书，如果是，则在要发往服务器的印章信息中写明“客户端openssl审核通过”，否则就标注“制作印章所使用的证书未能通过验证”。同时，客户端会将证书信息发送到服务器端进行二次验证。下拉提示信息框，可以看到服务器端的验证是否通过，如果服务器端的验证没有通过，是不能进行印章制作的。只后会从所使用的证书的私钥中提取出公钥，提取的方法是通过调用openssl的系统命令，生成相应公钥，在读取公钥，删除生成的公钥文件，待用户填写完制作印章的相关信息后就将全部信息写入印章。具体流程如下图所示。

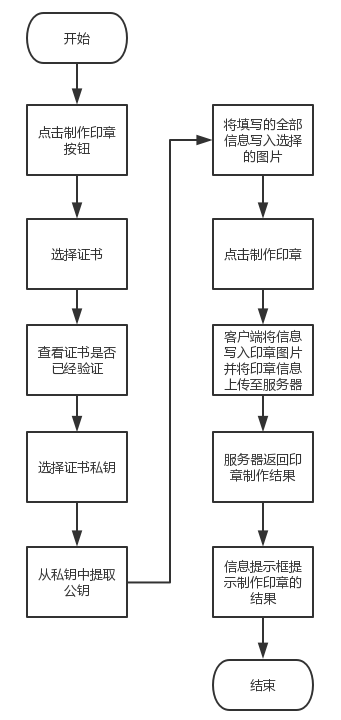


图5.2-1印章制作流程图

每一个生成的印章都使用了QT的QUuid::createUuid()方法生成了唯一标识码。

通过对PDF文档的分析，知道其是通过交叉引用表实现对文档间接对象的查找和读取，通过增量更新来实现对文档的修改。根据这两点，我们决定通过交叉引用表和文件尾获取到PDF文件的目录结构，通过获取到的目录和交叉引用表的结合获取要盖章页面的间接对象。然后通过重写页面间接对象的Contents属性和/XObject属性，将原Contents中的间接对象取出，用自己新建的间接对象代替，该间接对象用于存放原内容间接对象和新增的印章间接对象。在/XObject的字典中添加一项/Seal属性用于引用印章图片的间接对象，构造完这些间接对象后写一个增量更新的交叉引用表用于给PDF阅读器正常读取文件；之后在文件尾的%%EOF后另起一行写入印章的信息。由于印章信息并没有设置任何间接对向，不会对PDF文档本身造成影响，只要修改文件尾的文件总长度包含证书信息即可。之后会生成盖章后文件的SHA512码，点击提交按钮后，将该哈希码发送至服务器端使用PBKDF2算法加密后存入数据库中。由于不同的软件所生成的PDF文件内容的格式各有不同，且有些不按照Adobe公司的格式进行排版，所以本软件仅适配Adobe公司的PDF编辑、Microsoft的Word、WPS文字三款主流软件生成的PDF文件。其余软件生成的PDF文件不进行适配，这将作为后续的增值服务。其流程图如下图所示。

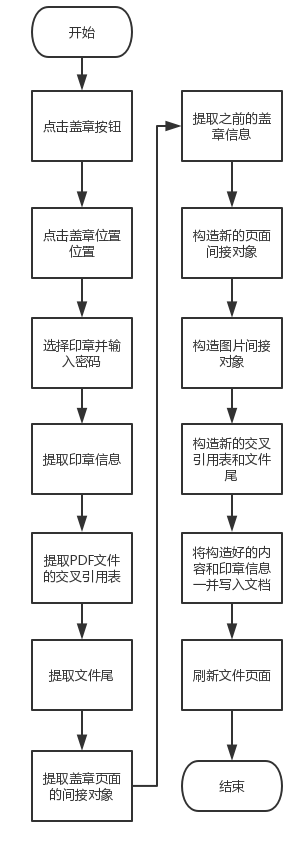


图5.2-2盖章流程图

在图5.2-2中并未加入提交文件哈希信息到服务器那一步，那步由用户自行决定是否进行，因为一旦提交就无法再进行撤章操作。在读取印章时，还会将印章的唯一标识码发送至服务器端进行校验，确定印章信息是否是已被记录在服务器端。这样涉及的目的是防止印章的复制，因为在上传盖章文件信息时会要求印章ID，然而印章ID是不会写到盖章文件中的，也就不可能上传复制印章的盖章文件信息至服务器端。

对于撤章功能则采用增量更新的方式实现，这样可以保证随意撤除任意印章，为了能修改印章的是否可读取编号，采用将文档信息和已有印章信息读取出来，修改后重新写会去，这也确保了盖章后的印章不能用增量更新的方式修改，如果用增量更新的方式修改，在撤章的时候就会被抹掉。该功能只能在还为将盖章文件信息上传至服务器端时使用

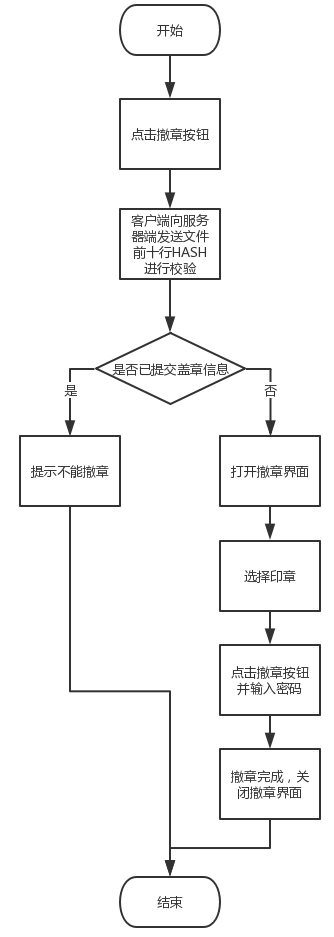


图5.2-3撤章流程图

如图5.2-3所示，提交的已盖章文件信息，使用文件的前十行的数据计算HASH值作为信息索引的键的，方便进行校验。

验证功能通过解析提取到的盖章增量更新，并将文件HASH信息发送至服务器校验实现。本地校验用于查看印章信息，和文件是否被修改，服务器端校验是为了检验文件信息是否已经提交保存至服务器，并获取盖章人、盖章人所属公司、印章名称、印章类型、印章信息，将其显示出来。其流程图如下图所示。

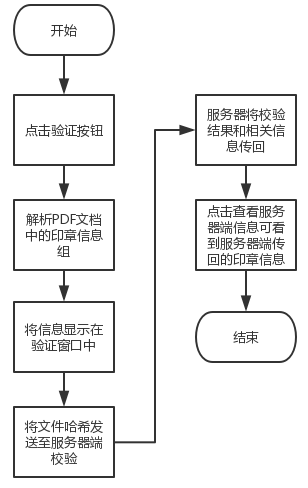


图5.2-4撤章流程图

### 5.2.2相关类及其描述

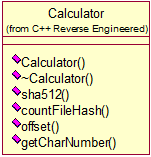
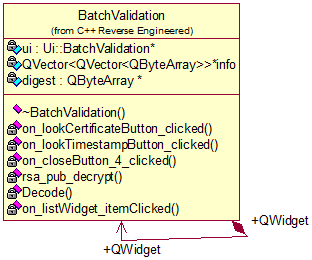
 

图5.2.2-1计算器类图 图5.2.2-2 验证页面类图

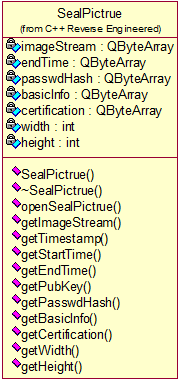
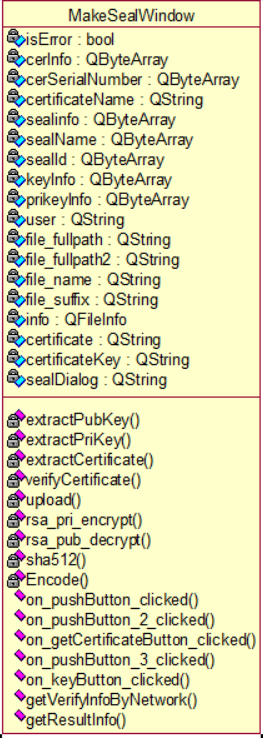
 

图5.2.2-3印章图片处理类图 图5.2.2-4印章制作页面类图

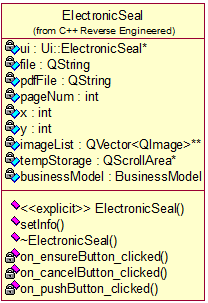
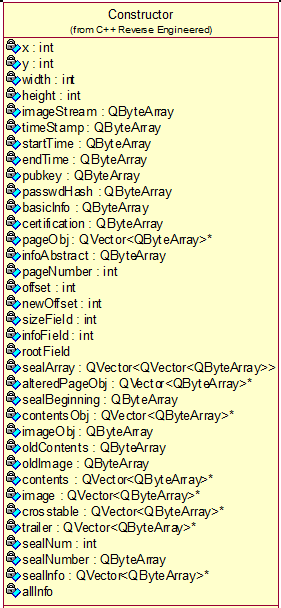
 

图5.2.2-5盖章页面类图 图5.2.2-6.1 印章数据构造器类图属性部分

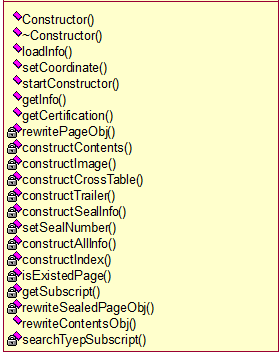


图5.2.2-6.2印章数据构造器类图方法部分

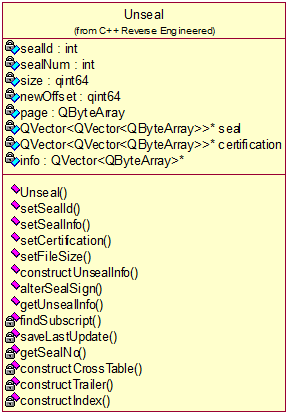
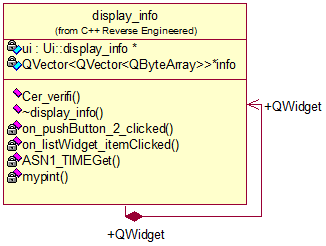


图5.2.2-7证书详细信息显示类图 图5.2.2-8撤章类类图

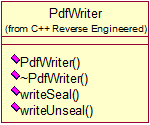


图5.2.2-9撤章和盖章信息写入类类图

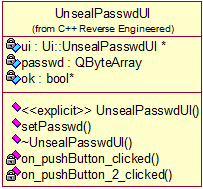
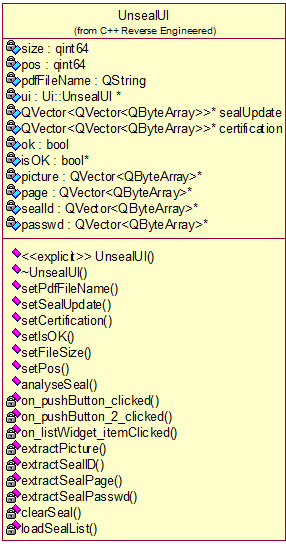
 

图5.2.2-10验证密码页面类类图 图5.2.2-11撤章页面类图

表5.2.2-1计算器类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| sha512 | const std::string | QByteArray | 计算传入字符串的哈希值 |
| countFileHash | QFile | QString | 计算文件的哈希值 |
| offset | Int,  int | Int | 计算文件偏移量 |
| getCharNumber | QVector<QByteArray>\* | Int | 计算字符串数组中的字符总数 |

表5.2.2-2验证页面类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| ui | 验证页面UI对象 |
| info | 传入的印章信息 |
| digest | 信息摘要 |

表5.2.2-3验证页面类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| on\_lookCertificateButton\_clicked | 无 | void | 查看证书按钮事件 |
| on\_lookTimestampButton\_clicked | 无 | void | 查看时间戳证书按钮事件 |
| on\_closeButton\_4\_clicked | 无 | void | 关闭按钮事件 |
| rsa\_pub\_decrypt | std::string，  std::string | std::string | rsa公钥解密,使用时传入密文和公钥信息即可 |
| Decode | const char\*，  Int，  int& | std::string | 解码方法 |
| on\_listWidget\_itemClicked | QListWidgetItem \* | void | 印章列表点击事件 |

表5.2.2-4印章图片处理类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| imageStream | 图片流 |
| timeStamp | 时间戳 |
| startTime | 起效时间 |
| endTime | 有效期截止时间 |
| pubKey | 公钥 |
| passwdHash | 密码哈希 |
| basicInfo | 基础信息 |

表5.2.2-5印章图片处理类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| openSealPictrue | openSealPictrue | bool | 打开印章图片并读取信息 |
| getImageStream | 无 | QByteArray | 获取印章图片数据流 |
| getTimestamp | 无 | QByteArray | 获取时间戳 |
| getStartTime | 无 | QByteArray | 获取印章起效时间 |
| getEndTime | 无 | QByteArray | 获取印章失效时间 |
| getPubKey | 无 | QByteArray | 获取印章公钥 |
| getPasswdHash | 无 | QByteArray | 获取私钥哈希值 |
| getBasicInfo | 无 | QByteArray | 获取基础信息 |
| getCertification | 无 | QByteArray | 获取印章信息 |
| getWidth | 无 | int | 获取图片的宽度 |
| getHeight | 无 | int | 获取图片的高度 |

表5.2.2-6印章制作页面类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| ui | Ui::MakeSealWindow类的实例 |
| file\_fullpath | 图片路径 |
| file\_fullpath2 | 图片路径 |
| file\_name | 图片名 |
| file\_suffix | 图片后缀 |
| info | 详细路径信息 |
| certificateKey | 证书秘钥 |
| sealDialog | 印章生成目录 |
| isError | 证书是否有错误 |
| cerInfo | 证书二进制数据 |
| cerSerialNumber | 证书序列号 |
| certificateName | 印章名 |
| sealinfo | 印章认证信息 |
| sealName | 印章名 |
| sealId | 印章id |
| keyInfo | 提取得到的公钥 |
| prikeyInfo | 私钥信息 |
| user | 印章制作人 |
| certificate | 证书 |

表5.2.2-7印章制作页面类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| on\_pushButton\_clicked | 无 | void | 确定按钮事件，开始制作印章 |
| on\_pushButton\_2\_clicked | 无 | void | 选择图片按钮，选择作为印章的图片 |
| on\_getCertificateButton\_clicked | 无 | void | 证书选择路径，选择用于签证的证书 |
| on\_pushButton\_3\_clicked | 无 | void | 选择印章生成路径的按钮 |
| on\_keyButton\_clicked | 无 | void | 选择证书私钥 |
| extractPubKey | 无 | void | 从私钥中提取公钥 |
| extractPriKey | 无 | void | 从私钥文件中提取私钥 |
| extractCertificate | 无 | void | 从证书文件中提取证书数据 |
| verifyCertificate | 无 | void | 验证证书 |
| upload | 无 | void | 上传印章信息 |
| rsa\_pri\_encrypt | const std::string &，  const std::string & | std::string | 私钥加密 |
| rsa\_pub\_decrypt | const std::string &，  const std::string & | std::string | 公钥解密 |
| sha512 | const std::string | std::string | 生成用户密码hash值 |
| Encode | const unsigned char\*，  int | std::string | base64加密 |

表5.2.2-8盖章页面类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| ui | Ui::ElectronicSeal的实例化对象 |
| file | 印章文件路径 |
| pdfFile | PDF文件路径 |
| pageNum | 要盖章页面的页码 |
| x | 盖章位置的横坐标 |
| y | 盖章位置的纵坐标 |
| imageList | 缓存的PDF文档图片列表 |
| tempStorage | 文档显示区域 |
| businessModel | BusinessModel类的实例对象 |

表5.2.2-9盖章页面类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| setInfo | QString，  QVector<QImage>\*\*，  QScrollArea\*，  Int，  Int，  int | void | 装载要盖章的文档的地址、盖章页面图片缓存、文档显示区的信息、盖章位置的信息、盖章页码 |
| on\_ensureButton\_clicked | 无 | void | 确认按钮，点击后进行密码验证并盖章 |
| on\_cancelButton\_clicked | 无 | void | 取消按钮 |
| on\_pushButton\_clicked | 无 | void | 印章选择按钮 |

表5.2.2-10印章数据构造器类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| x | 印章在pdf中的x坐标 |
| y | 印章在pdf中的y坐标 |
| width | 印章宽度 |
| height | 印章高度 |
| imageStream | 图片流 |
| timeStamp | 时间戳 |
| startTime | 起效时间 |
| endTime | 有效期截止时间 |
| pubKey | 公钥 |
| passwdHash | 密码哈希 |
| basicInfo | 基础信息 |
| certification | 印章证书 |
| pageObj | 要盖章的页面的间接对象 |
| infoAbstract | 文件信息摘要 |
| pageNumber | 盖章页面编号 |
| offset | 原文件偏移量 |
| newOffset | 新构造完成后的文件偏移量 |
| sizeField | 原pdf文件的对象总数 |
| infoField | 原pdf文件的信息对象标号 |
| rootField | 原pdf文件的root对象的编号 |
| sealArray | 已有印章信息 |
| alteredPageObj | 修改后的页面对象 |
| sealBeginning | 印章开头 |
| contentsObj | 新的内容对象 |
| imageObj | 新的图片对象 |
| oldContents | 旧内容对象 |
| oldImage | 旧图片对象 |
| contents | 要添加的新内容间接对象 |
| image | 要添加的图片间接对象 |
| crosstable | 交叉引用表 |
| trailer | 文件尾 |
| sealNum | 原印章数 |
| sealNumber | 盖章后总印章数 |
| sealInfo | 根据印章图片的信息重新组合 |
| allInfo | 构建完整的增量更新信息 |

表5.2.2-11印章数据构造器类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| loadInfo | SealPictrue\*,  QVector<QByteArray>\*,  QByteArray,  Int,  Int,  int  Int,  Int,  QVector<QVector<QByteArray>>\* | bool | 信息装载 |
| setCoordinate | Int,  int | bool | 设置盖章位置 |
| startConstructor | 无 | bool | 开始构造信息 |
| getInfo | 无 | QVector<QByteArray>\* | 获取构造完成后的信息 |
| getCertification | 无 | QVector<QByteArray>\* | 获取证书信息 |
| rewritePageObj | 无 | void | 重写页面对象 |
| constructContents | 无 | void | 构造内容对象 |
| constructImage | 无 | void | 构造图片对象 |
| constructCrossTable | 无 | void | 构造交叉引用表 |
| constructTrailer | 无 | void | 构造文件尾 |
| constructSealInfo | 无 | void | 构造印章信息 |
| setSealNumber | 无 | void | 设置印章个数 |
| ConstructAllInfo | 无 | void | 构造完整的增量更新 |
| constructIndex | int | void | 交叉引用表索引构造 |
| isExistedPage | 无 | bool | 判断是否为已盖章页面 |
| getSubscript | 无 | int | 查找盖章页面最后一次增量更新在印章Vector中的位置 |
| rewriteSealedPageObj | int | void | 重写盖章页面 |
| rewriteContentsObj | QVector<QByteArray>\* | void | 重写盖章页面的间接内容对象 |
| searchTyepSubscript | 无 | int | 查找Type下标 |

表5.2.2-12证书详细信息显示类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| ui | Ui::display\_info 类的实例化对象 |
| info | 印章证书信息 |

表5.2.2-13证书详细信息显示类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| Cer\_verifi | const char \* | void | 验证x509证书使用时传入证书内容即可 |
| ASN1\_TIMEGet | ASN1\_TIME\*,  int\* | time\_t | 用于解析证书的起始时间和截止时间，将ASN1\_TIME转为time\_t |
| Mypint | unsigned const char\*\*,  Int,  Int,  Int,  Int\* | int | 用于转换时间 |
| on\_pushButton\_2\_clicked | 无 | void | 关闭按钮 |
| on\_listWidget\_itemClicked | QListWidgetItem \* | void | 证书列表点击事件 |

表5.2.2-14撤章类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| sealId | 印章编号 |
| sealNum | 印章数目 |
| size | PDF文件大小 |
| newOffset | 添加撤章增量更新后新的PDF文件偏移量 |
| page | 印章所在页面 |
| seal | 印章增量更新列表 |
| certification | 证书信息列表 |
| info | 用于存储要撤的章所在页面的最后一次增量更新 |
| unsealInfo | 构造好的撤章增量更新 |

表5.2.2-15撤章类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| setSealId | int | void | 设置要删去的印章的ID |
| setSealInfo | QVector<QVector<QByteArray>>\* | void | 装载印章信息列表 |
| setCertification | QVector<QVector<QByteArray>>\* | void | 装载证书信息列表 |
| setFileSize | qint64 | void | 装载文件字符总数（总字节数） |
| constructUnsealInfo | 无 | bool | 构造撤章增量更新 |
| alterSealSign | 无 | void | 修改印章标记 |
| getUnsealInfo | 无 | QVector<QVector<QByteArray>> \* | 获取构造完毕的撤章增量更新 |
| findSubscript | 无 | int | 找到页面最后一次增量更新的地方 |
| saveLastUpdate | int | void | 存储最后一次增量更新 |
| getSealNo | int | int | 获取印章在盖章页面的序号 |
| countSeal | int | void | 计算在某一页面的印章个数 |
| constructCrossTable | 无 | void | 构造交叉引用表 |
| constructTrailer | 无 | void | 构造文件尾 |
| constructIndex | qint64 | void | 交叉引用表索引构造 |

表5.2.2-16撤章和盖章信息写入类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| writeSeal | QString，  QVector<QByteArray>\* | bool | 向PDF文件写入印章增量更新 |
| unsealInfo | QString，  QVector<QVector<QByteArray>>\* | bool | 向PDF文件写入撤章增量更新 |

表5.2.2-17验证密码页面类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| ui | Ui::UnsealPasswdUI \*的实例化对象 |
| passwd | 传入的印章密码 |
| ok | 密码是否正确的判断 |

表5.2.2-18验证密码页面类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| setPasswd | QByteArray，  bool\* | void | 装载密码 |
| on\_pushButton\_clicked | 无 | void | 确定按钮事件，确定即开始验证 |
| on\_pushButton\_2\_clicked | 无 | void | 取消按钮事件 |

表5.2.2-19撤章页面类属性表

|  |  |
| --- | --- |
| 属性名 | 描述 |
| size | PDF文件大小 |
| pos | 第一次印章增量更新前的文件光标位置 |
| pdfFileName | PDF文件名及所在目录 |
| ui | Ui::UnsealUI \*的实例化对象 |
| sealUpdate | pdf中已存在印章操作更新 |
| certification | pdf中的证书信息 |
| ok | 验证密码是否正确 |
| isOK | 判断是否撤章成功 |
| picture | 印章图片 |
| page | 印章所在页面 |
| sealId | 印章在PDF文件中的编号 |
| passwd | 印章密码 |

表5.2.2-20撤章页面类方法表

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 接收参数类型  (按先后顺序排列) | 返回类型 | 描述 |
| setPdfFileName | QString | void | 传入文件名 |
| setSealUpdate | QVector<QVector<QByteArray>>\* | void | 传入印章增量更新 |
| setCertification | QVector<QVector<QByteArray>>\* | void | 传入证书信息列表 |
| setIsOK | bool\* | void | 传入判断是否盖章成功标记 |
| setFileSize | qint64 | void | 传入文件大小 |
| setPos | qint64 | void | 传入第一次印章增量更新前的文件光标位置信息 |
| analyseSeal | 无 | void | 解析传入的印章列表中的信息 |
| on\_pushButton\_clicked | 无 | void | 撤章按钮事件，点击后会弹出密码验证请求 |
| On\_pushButton\_2\_clicked | 无 | void | 取消按钮，关闭撤章页面 |
| on\_listWidget\_itemClicked | QListWidgetItem \* | void | 印章列表，点击可查看对应印章信息 |
| extractPicture | 无 | void | 提取印章图片数据 |
| extractSealID | 无 | void | 提取印章在PDF文件中的编号 |
| extractSealPage | 无 | void | 提取印章所在页面页码 |
| extractSealPasswd | 无 | void | 提取印章密码的哈希值 |
| clearSeal | int | void | 清除印章增量更新 |
| loadSealList | 无 | void | 装载有效印章列表并显示。 |

### 5.2.3核心代码

#### 5.2.3.1盖章模块

pdfdocument.cpp

PdfUtils\* pdfUtils = new PdfUtils(page);

pdfUtils->setSealInfo(sealUpdate);

pdfUtils->openSealFile(pdfFile);

Constructor\* constructor = new Constructor();

constructor->loadInfo(seal, pdfUtils->getPageObj(), pdfUtils->getInfoAbstract(),page, pdfUtils->getOffset(), pdfUtils->getSize(),pdfUtils->getInfo(), pdfUtils->getRoot(), sealUpdate);

constructor->setCoordinate(x, y);

constructor->startConstructor();

PdfWriter\* pdfWriter = new PdfWriter();

pdfWriter->writeSeal(pdfFile, constructor->getInfo());

(\*certification).append(\*constructor->getCertification());

(\*sealUpdate).append(\*constructor->getInfo());

pdfUtils->openPdf(pdfFile);

pdfUtils->getPdfImage(page, imageList);

PdfCanvas\* pdfCanvas = new PdfCanvas();

pdfCanvas->drawCanvas(imageList,tempStorage,page);

delete pdfCanvas;

pdfUtils->freeResource();

delete pdfUtils;

delete constructor;

#### 5.2.3.2撤章模块

businessmodel.cpp

bool\* isOK = new bool;

\*isOK = false;

UnsealUI\* unseal = new UnsealUI();

unseal->setPdfFileName(pdfFileName);

unseal->setSealUpdate(sealUpdate);

unseal->setCertification(certification);

unseal->setFileSize(\*pdfSize);

unseal->setIsOK(isOK);

unseal->setPos(\*pos);

unseal->analyseSeal();

unseal->exec();

if(isOK)

{

PdfDocument\* document = new PdfDocument();

document->unseal(imageList, tempStorage, pageNum, pdfFileName);

}

pdfdocument.cpp

pdfUtils = new PdfUtils();

pdfUtils->openPdf(pdfFileName);

pdfUtils->getPdfImage(pageNum, imageList);

PdfCanvas\* pdfCanvas = new PdfCanvas();

pdfCanvas->drawCanvas(imageList,tempStorage,pageNum);

delete pdfCanvas;

delete pdfUtils;

#### 5.2.3.3制章模块

makesealwindow.cpp

......

QString syscmd ="openssl x509 -req -in " + sealDialog + "/seal.csr -out " + sealDialog + "/" + name+ ".crt -signkey " + sealDialog + "/prikey.key -CA "+ certificateDialog +"/\*.crt -CAkey "+ certificateDialog +"/\*.key -CAcreateserial -days " + effectiveTime;

QString createSealCertificate = "openssl req -utf8 -new -key " + sealDialog + "/prikey.key -out " + sealDialog + "/seal.csr -nodes -subj \"/C=" + country + "/ST=" + province + "/L=" + location + "/O=" + company + "/OU=" + user + "/CN=" + name + "\"";

......

//基本信息组合，进行RSA加密

QString information = company + "/" + user + "/" + name + "/" + pwd + "/" + startime\_stmp2 + "/" +stardate + "/" + endtime ;

std::string str\_information = information.toStdString();

generateRSAKey(key);

system(str\_csc);// 根据私钥文件，为应用证书/中级证书生成 csr 文件（证书请求文件）

system(cmd);//给用户私钥颁发证书

QFile::copy(file\_fullpath, file\_fullpath2);

QString certification = sealDialog + "/" + name+ ".crt";

ifstream pdfCertification;

ofstream outFile;

pdfCertification.open(certification.toStdString());

outFile.open(file\_fullpath2.toStdString(),ios::app);

outFile << endl<< "\_\_Delimiter\_\_" << endl;

outFile << startime\_stmp << endl; //签发时间戳

outFile << str\_stardate << endl; //签发时间

outFile << str\_endtime << endl; //印章有效期

outFile << "\_\_Delimiter\_\_" << endl;

outFile << key[0]; //公钥

outFile << "\_\_Delimiter\_\_" << endl;

outFile << sha512(str\_pwd) << endl; //用户密码hash

outFile << "\_\_Delimiter\_\_" << endl;

outFile << rsa\_pri\_encrypt(str\_information, key[1]) << endl;//使用私钥对基本信息RSA加密（签名）

outFile << "\_\_Delimiter\_\_" << endl;

char ch;

while(pdfCertification.get(ch))

{

outFile<<ch;

}

outFile.close();

#### 5.2.3.4验证模块

Batchvalidation.cpp

for(int i = 0;i<n;i++)//在widget中画出相应的电子印章

{

item<<"电子印章"+QString::number(i+1);

}

ui->listWidget->addItems(item);

ui->sealNumber->setText(QString().number(n));

QVector<QByteArray> verifi;

std::string aa;

QByteArray verti,pubkey,ciphertext,Info\_digest;//分别为单个印章信息，公钥，加密后的密文，证书,信息摘要

for(int i = 0;i < n;i++) //读出电子签章内的印章信息，公钥和加密后的密文

{

verifi = this->info->at(i);

k = 0;

for(int j = 0;j<verifi.size();j++)

{

verti = verifi.at(j);

for(int t = 0;t<verti.size();t++)

{

if(verti.at(t) == '\n'){

k++;

}

if(k >=5 && k<=12){

pubkey += verti.at(t);

}

else if(k >= 13 && k <= 17){

ciphertext += verti.at(t);

}

else if(k == 4){

Info\_digest += verti.at(t);

}

}

}

qDebug()<<pubkey<<ciphertext;

aa = Decode((const char \*)ciphertext.data(),ciphertext.length(),i);//base64解码

aa = rsa\_pub\_decrypt(aa,pubkey.data()); //rsa公钥解密

pubkey.clear();

ciphertext.clear();

if(this->digest == Info\_digest){ //无法识别的印章，即信息摘要无法验证错误的印章

unverfi++;

}

Info\_digest.clear();

if(aa.empty()) unsuc++;

}

ui->validSeal->setText(QString().number(n-unsuc-unverfi));

ui->invalidSeal->setText(QString().number(unsuc));

ui->indistingguishableness->setText(QString().number(unverfi));

## 网络相关操作模块设计

### 5.3.1思路

该部分所涉及的各模块均基于QT的QNetworkAccessManager实现，主要是通过Http协议进行数据的提交和申请。

这之中的登陆模块的实际实现方式就是通过Http协议向服务器端提交账号密码，服务器端验证后返回用户的邮箱、权限和昵称。为了实现类间的通信，采用QT特有的槽函数机制，当接收到服务器传回的信息后自动调用相应的处理方法，并传给发送申请的类。

### 5.3.2相关类及其描述

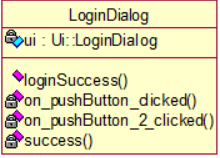
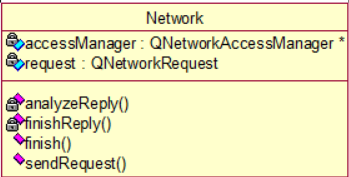
 

图5.3.2-1登陆界面类类图 图5.3.2-2网络类类图

由于登陆界面类与网络类的属性和方法较少，所以采用直接描述的方法进行介绍，不再列出表格。

对于LoginDialog类，其唯一的属性为界面实例。loginSuccess()方法为信号，在登陆成功时触发，触发后与该信号绑定的槽函数就会触发。两个按钮的绑定槽函数分别对应登陆和注册。success()方法为槽函数，在服务器返回用户信息时被槽函数机制调用。

对于Network类，第一个参数为HTTP协议的管理器，用于提交Http请求和在收到服务器短信息时触发finishReply方法转发接收到的数据到向服务器发送请求的类中。finish()方法是信号，用于传递服务器传回信息，接收的参数为QJsonObject类型。sendRequest()方法用于向服务器发送数据，传入的参数为要传的数据以及服务器域名。analyzeReply用于解析服务器传回的JSON格式数据。

## 网页端客户上传信息模块

### 5.4.1思路

文件上传主要靠element的el-upload控件实现，利用该控件实现文件的上传，在服务器部分接收到前端提交的文件后将其保存到静态资源文件夹里，然后再把用户的信息以及上传的所有信息一起上传到数据库中去。

### 5.4.2相关类及描述

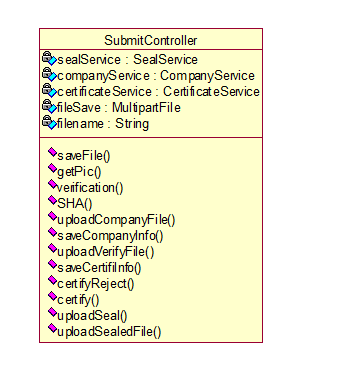
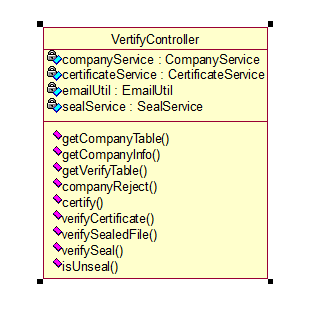
 

图5.4.2-1公司信息控制类图 图5.4.2-2证书信息控制类图

### 5.4.3核心代码

public JSONArray employees(String state) throws JSONException {

JSONArray jsonArray = new JSONArray();

List<Users> user = userinfoRepository.employees(state);

for (Users userInfo : user) {

JSONObject json = new JSONObject();

json.put("id", userInfo.getId());

json.put("username", userInfo.getUsername());

json.put("email", userInfo.getEmail());

json.put("department", userInfo.getDepartment());

json.put("cellphone", userInfo.getCellphone());

json.put("power", userInfo.getPower());

json.put("isverify", userInfo.getIsverify());

jsonArray.put(json);

}

return jsonArray;

}

/\*

删除用户√

\*/

public JSONObject delete(String email) throws JSONException {

JSONObject json = new JSONObject();

Users user = userinfoRepository.getByEmail(email);

if (user == null) {

json.put("error", "不存在该用户");

} else {

userinfoRepository.delete(email, "已注销");

json.put("result", "删除成功");

}

return json;

}

/\*

添加用户√create(account,password,name,compan.getId(),phone,attribute.getId()).toString();

\*/

public JSONObject create(String username, String passwd, String email, String company, String cellphone, String power, String isverify, String state) throws JSONException {

JSONObject json = new JSONObject();

Users user = userinfoRepository.getByEmail(email);

if (user != null) {

json.put("error", "已存在该用户");

} else {

user = new Users();

user.setUsername(username);

user.setPasswd(passwd);

user.setEmail(email);

user.setDepartment(company);

user.setCellphone(cellphone);

user.setPower(power);

user.setIsverify(isverify);

user.setState(state);

userinfoRepository.insert(user);

json.put("result", "添加成功");

}

return json;

}

/\*修改用户信息

\*

\*/

public JSONObject changeInfo(String email, String username, String cellphone, String company, String power) throws JSONException {

JSONObject json = new JSONObject();

Users userInfo = userinfoRepository.getByEmail(email);

if (userInfo == null) {

json.put("error", "该用户不存在");

return json;

}

Users user = new Users();

userInfo.setEmail(email);

userInfo.setUsername(username);

userInfo.setCellphone(cellphone);

userInfo.setDepartment(company);

userInfo.setPower(power);

userinfoRepository.updateAccountInfo(userInfo);

json.put("result", "修改成功");

return json;

}

/\*

修改密码

\*/

public JSONObject changePassword(String email, String oldPassword, String newPassword) throws JSONException {

JSONObject json = new JSONObject();

Users user = userinfoRepository.getByEmail(email);

if (user == null) {

json.put("error", "用户不存在");

} else if (oldPassword.equals(user.getPasswd())) {//校验旧密码是否正确

user.setPasswd(newPassword);

user.setEmail(email);

userinfoRepository.updatePassword(user);

json.put("result", "修改密码成功");

} else {

json.put("error", "原密码错误");

}

return json;

}

//获取当前用户信息

public JSONObject info(String account) throws JSONException {

JSONObject json = new JSONObject();

Users user = userinfoRepository.getByEmail(account);

if (user == null) {

json.put("error", "您查找的用户不存在");

} else {

json.put("id", user.getId());

json.put("username", user.getUsername());

json.put("email", user.getEmail());

json.put("cellphone", user.getCellphone());

}

return json;

}

public JSONObject loginClient(String account, String password) throws JSONException, NoSuchAlgorithmException, InvalidKeySpecException {

JSONObject object = new JSONObject();

Users users;

users = userinfoRepository.getByEmail(account);

if (users == null) {

object.put("error", "该用户不存在");

return object;

} else {

if (!PBKDF2.verify(password, PBKDF2.getSalt(), users.getPasswd())) {

object.put("username", users.getUsername());

object.put("email", users.getEmail());

object.put("department", users.getDepartment());

object.put("result", "普通用户");

} else {

object.put("error", "密码输入错误");

}

}

return object;

}

## 5.5 网页端管理员审核信息模块

### 5.5.1思路

对于信息的审核主要为了保证用户上传的证书以及公司的信息是确实可靠的，从而达到保证用户电子印章以及PDF文档的可靠性，对于审核需要分成两个部分进行，公司的信息审核需要给审核人员查看公司的名称、社会系用代码、营业执照等，审核人员点击审核表的详情按钮后就将审核信息显示给审核人员查看，随后确认这些信息的有效性后选择确认或者驳回。

而对于证书信息的审核，需要审核人员将审核信息下载到自己的电脑上去然后使用电脑自带的证书查看软件进行证书的查看。确认证书的信息真实有效果后可以选择对证书进行确认已经驳回。下载文件利用文件存储位置的url实现。

### 5.5.2相关类及描述

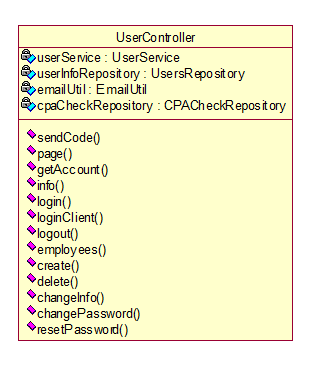
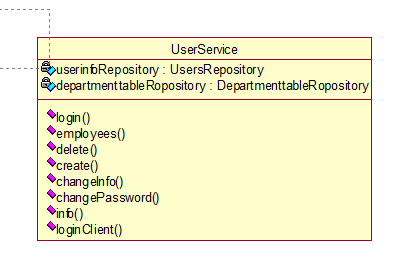
 

图5.5.2-1用户控制类图 图5.5.2-2用户服务类图

### 5.5.3核心代码

public void saveFile(String path, MultipartFile file) throws IOException {

OutputStream os = null;

InputStream inputStream = null;

String fileName = null;

try {

inputStream = file.getInputStream();

fileName = file.getOriginalFilename();

} catch (IOException e) {

e.printStackTrace();

}

try {

// 2、保存到临时文件

// 1K的数据缓冲

byte[] bs = new byte[1024];

// 读取到的数据长度

int len;

// 输出的文件流保存到本地文件

File tempFile = new File(path);

if (!tempFile.exists()) {

tempFile.mkdirs();

}

os = new FileOutputStream(tempFile.getPath() + File.separator + fileName);

// 开始读取

while ((len = inputStream.read(bs)) != -1) {

os.write(bs, 0, len);

}

} catch (IOException e) {

e.printStackTrace();

} catch (Exception e) {

e.printStackTrace();

} finally {

// 完毕，关闭所有链接

try {

os.close();

inputStream.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}

/\*

上传已盖章文件

\*/

@RequestMapping("/uploadVerification")

public String getPic(HttpServletRequest request, MultipartFile file) throws IOException {

fileSave = file;

return "1";

}

/\*

验证文件中电子印章的可靠性

\*/

@RequestMapping("/verification")

public String verification(@RequestParam(value = "id", required = false) String id) throws IOException {

String label = null;

int result;

//读出文件的前十行

fileSave.getResource().getInputStream();

File sealedFile = fileSave.getResource().getFile();

BufferedReader br = new BufferedReader(new FileReader(sealedFile));

String line = null;

for (int i = 0; i < 10; i++) {

line += br.readLine();

}

result = sealService.findLabel(id, SHA(line, "SHA-512"));

return String.valueOf(result);

}

private String SHA(final String strText, final String strType) {

// 返回值

String strResult = null;

// 是否是有效字符串

if (strText != null && strText.length() > 0) {

try {

// SHA 加密开始

// 创建加密对象 并傳入加密類型

MessageDigest messageDigest = MessageDigest.getInstance(strType);

// 传入要加密的字符串

messageDigest.update(strText.getBytes());

// 得到 byte 類型结果

byte[] byteBuffer = messageDigest.digest();

// 將 byte 轉換爲 string

StringBuilder strHexString = new StringBuilder();

// 遍歷 byte buffer

for (byte b : byteBuffer) {

String hex = Integer.toHexString(0xff & b);

if (hex.length() == 1) {

strHexString.append('0');

}

strHexString.append(hex);

}

// 得到返回結果

strResult = strHexString.toString();

} catch (NoSuchAlgorithmException e) {

e.printStackTrace();

}

}

return strResult;

}

# 软件使用测试

使用客户端前，首先需要到网站去注册账号和提交证书进行审核，没有通过审核的证书是不能用于制作印章的，没有印章就无法进行盖章等操作。如果以游客身份登录，只能使用客户端的文档查看功能，其余功能均无法使用。



图6-1登录框

如图6-1所示，如果已经注册输入账号密码点击登录即可，想使用游客方式登录就直接关闭登录窗口即可。点击注册会跳转至网站首页，在网站完成相关操作后，就可以正常使用客户端了。

登录后，会进入主界面，主界面的左下角会显示登录用户的昵称，如下图所示。

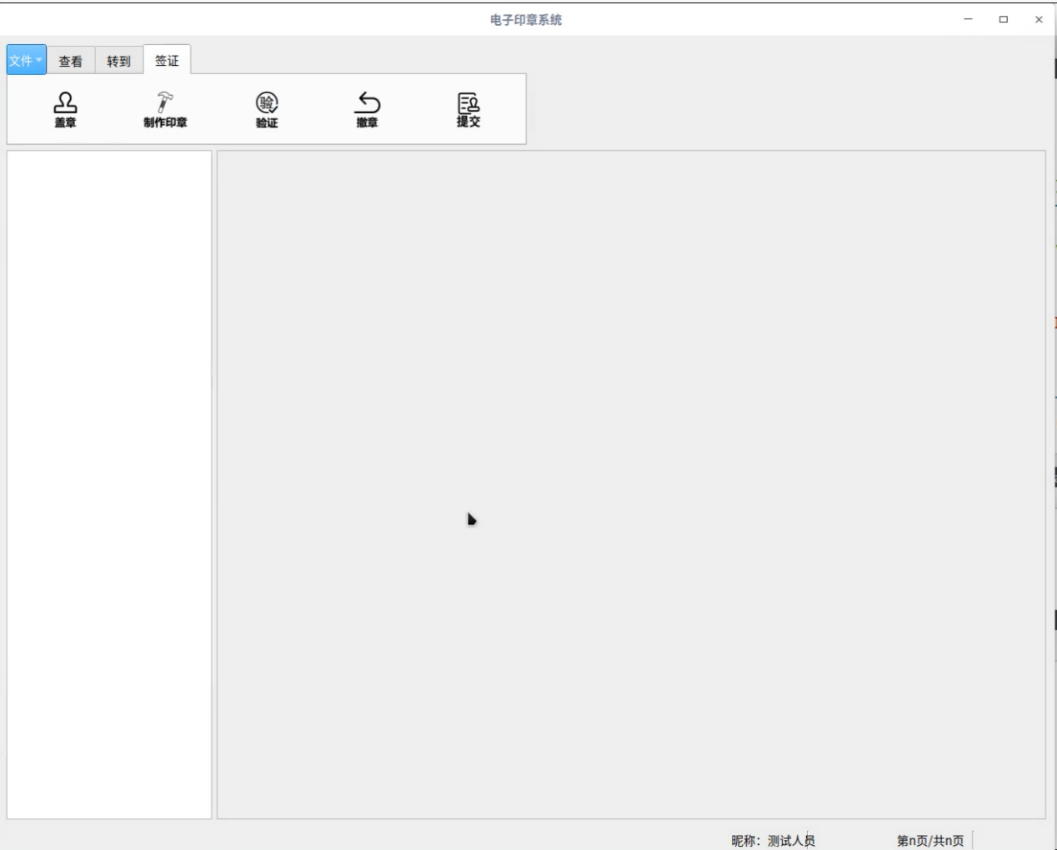


图6-2登录后的主界面

如图6-2所示，主界面的左下角的昵称处多出了登录用户的名称，这里使用的是昵称为测试人员的用户。

## 6.1 PDF文档操作模块

### 6.1.1打开文件

进入软件后用户点击打开文件按钮即可选择PDF文件所在位置。

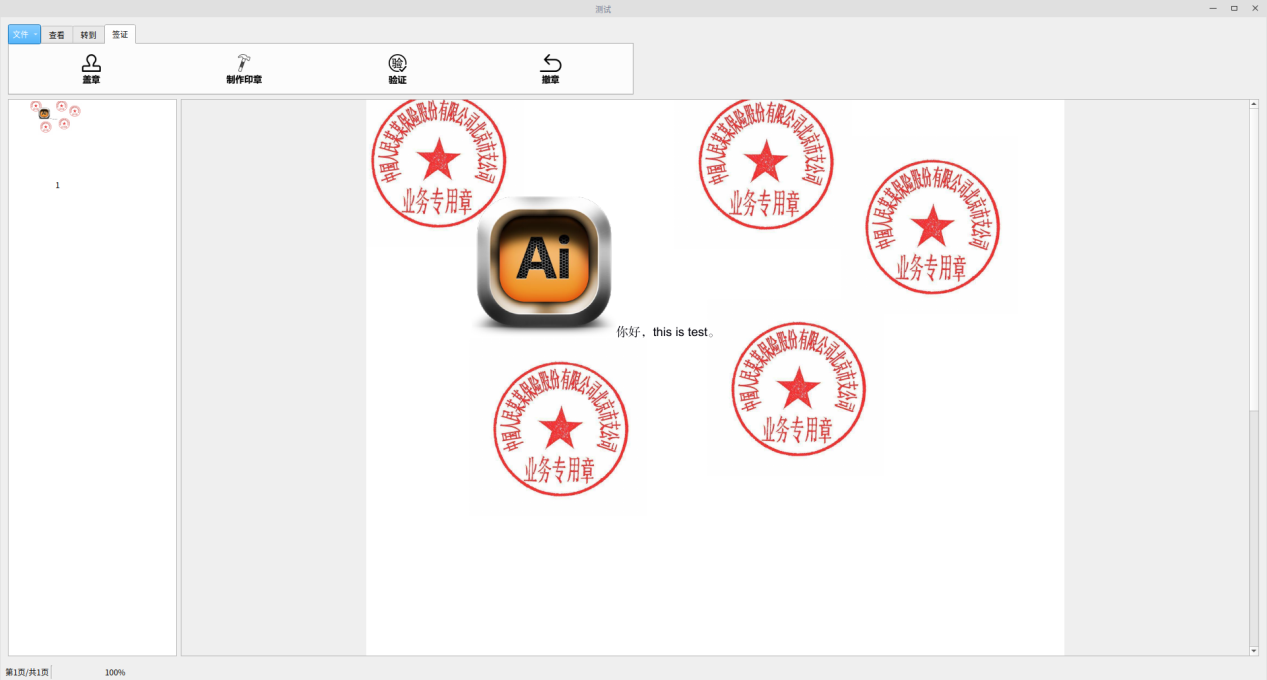


图6.1.1-1文档打开成功。

## 6.2 电子印章操作模块

### 6.2.1盖章



图6.2.1-1 电子印章选择界面

图中为选择盖章功能并选择盖章位置后对用户的提示，用户需要选择电子印章以及输入印章密码。输入完毕后点击确定继续。

  
图6.2.1-2图中显示为盖章成功后的主界面。

### 6.2.2撤章



图6.2.2-1 撤章页面

该图为撤章界面，当用户选择相应的电子印章后会出现该电子印章的预览与该电子印章所在页面与相关信息。点击撤章后进入下一步操作

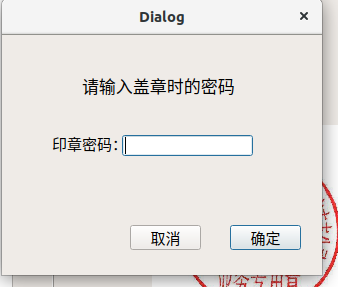


图6.2.2-2撤章密码验证

该图为撤章验证用户信息界面，用户只有输入电子印章密码并点击确认后才能成功进行撤章。

### 6.2.3制章



图6.2.3印章制作界面

按照图中提示输入相关信息，点击制作即可制作电子印章，点击制章后会提示制章成功信息。注意：目前只接受.crt类型的证书和证书秘钥，两者要放在一起。

### 6.2.4验证



图6.2.4-1印章验证界面

该图为批量验证界面，选择电子印章后会对电子印章进行相关的验证，并且提供查看时间戳以及查看证书功能。

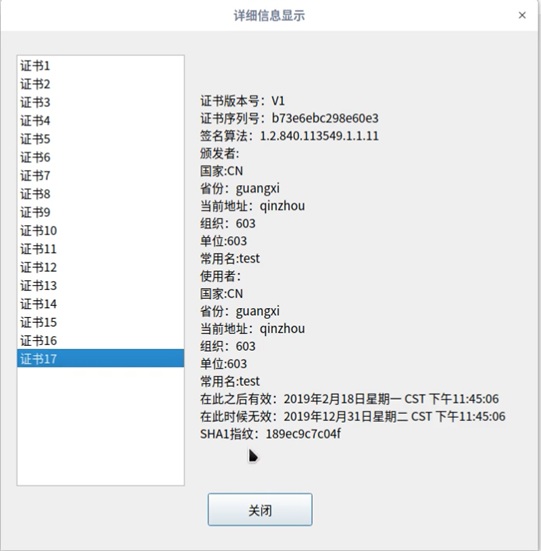


图6.2.4-2证书信息查看页面

如图6.2.4-2所示，第二部分的证书显示界面显示印章的证书的相关信息。时间戳证书的界面布局与证书信息显示界面的布局相类似，所以不再重复贴图。

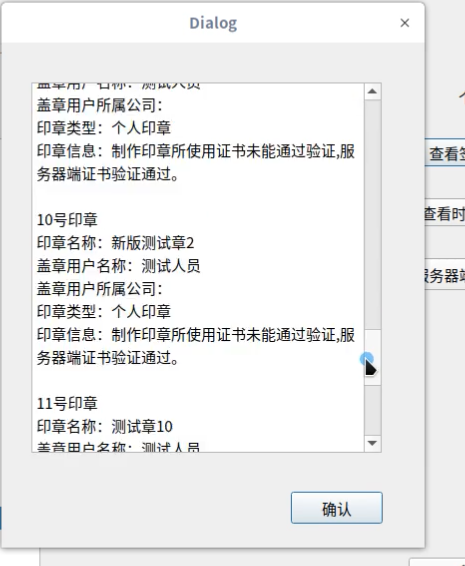


图6.2.4-3服务器端回传信息显示界面

如图6.2.4-3所示，第三部分的服务器端回传信息显示界面显示的是盖章用户名称、盖章用户所属公司，印章类型和印章信息。用户在还没进行企业认证前是没有所属公司的，同样的也不可以制作企业印章而只能制作个人印章。印章信息用于显示该印章的证书验证信息。整体通过一个下拉文本框按行显示，确认按钮用于关闭页面。

## 6.3服务器端相关模块



图6.3-1网站主页

如图所示用户打开网页后通过页面上方的菜单导航栏进行页面跳转登录页面进行登录



图6.3-2登录页面

在登录页面输入用户的邮箱以及密码后点击登录按钮进行登录，用户在网页端能够使用的功能主要有以下几个：查看软件的相关信息以及上传用户的公司信息和证书信息。在主页菜单栏点击菜单进行页面跳转开始上传信息的输入，如下图所示：



图6.3-3证书提交页

将所有需要的信息输入完成后点击确定即可等待管理员进行认证根据提示上传相应文件即可。同时新用户也可以选择点击注册功能进行新用户的注册，在新用户的注册过程中需要保证用户输入的邮箱是未注册过的邮箱，在输入邮箱后点击获取验证码即可通过邮箱得到服务器发送的验证码。用户将验证码以及相关的信息输入完毕即可完成注册。

而管理员的主要功能有查看用户信息修改用户信息以及审核用户上传信息。管理员在登陆后的初始界面中即可查看用户的信息，如下图所示：



图6.3-4用户管理页面

管理员可以点击表格后方的操作按钮对用户的信息进行修改或者注销用户，当用户选择修改用户信息后会出现如下页面，其中包括修改用户的账号名字等基本信息，管理员在确认身份信息无误后可以进行修改。如下图所示：



图6.3-5用户信息修改页

而审核信息则是通过列表的方式来显示当前所有需要审核的信息，管理员可以通过操作按钮查看公司审核信息的相关细节，再确认细节无误后可以选择通过或者驳回，如下图所示：



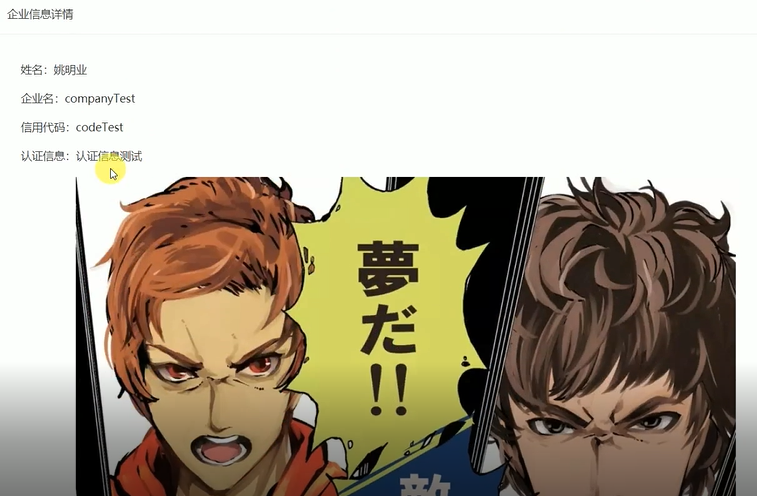


图6.3-6企业信息审核页

而对于证书信息的审核，管理员可以通过操作中的下载按钮进行证书文件的下载，将文件下载到本地后进行查看，确认后可以选择通过或者驳回操作。如图所示：



图6.3-7证书信息审核页



图6.3-8证书下载

# 结束语

## 7.1总结

### 7.1.1已完成的功能模块

目前已完成的功能模块有PDF文档的打开功能、文档目录的加载功能、文档的页面跳转功能、文档页面的缩放功能、基于openssl的x509印章制作和印章验证功能，盖章功能。服务器端企业信息认证和证书信息认证功能，用户信息存储及用户登陆功能，印章信息保存功能、盖章文件保存功能，证书保存功能。

### 7.1.2使用价值

本软件使得Linux系统也有了相应的PDF文档的电子印章制作及应用系统，由于印章的写入方法，与Adobe公司的规范不同，使用其他公司的PDF阅读器只能看到印章，而不能对印章进行任何操作，只有使用本软件才能对印章进行验证和撤除。并且一旦使用本软件在PDF文档中进行盖章，之后对文档的盖章页面的内容对象的修改，会使印章无法显示。撤章的时候，会将在盖章后进行的增量更新全部清除，只保留印章信息和撤章增量更新。由于是使用增量更新的方式进行撤章，所以可撤除任意位置的印章。

将印章及用户的证书信息以及企业信息相关联，保存在服务器端，既方便查询，也能保证不会被其他人员对pdf中的印章进行篡改和复制。与传统的单客户端形式的pdf文件盖章工具相比具有更高的安全性，不易被外部人员破解。不会因为软件印章制作和盖章功能被破解就导致文档印章被篡改和复制。且目前市面上还没有相同类型的产品。

## 7.2展望

### 7.2.1未完成的功能模块

目前未完成的功能有书签功能、查询功能、打印功能，对非本软件所盖章和数字签名的验证的功能，以png图片作为印章图片实现透明背景的印章的功能还未实现，即目前本软件只能使用jpg和jpeg图片文件作为印章图片，切两个印章间如果挨得过近会相互遮挡影响美观。

### 7.2.2下一步的打算

实现未完成功能，开发软件的进阶版即区块链版本。现阶段的电子印章应用软件通常都是靠闭源的方式隐藏其实现盖章功能的方法，因为其一旦被人得知，很快就会被破解，导致印章被肆意复制、修改，其安全性也就不复存在了。由于是闭源，只有软件开发公司知道具体的实现方法，一旦开发公司中有人心生歹念，利用其对软件的理解，肆意复制印章并从中谋利，将会导致严重损失。而使用区块链技术就可以解决这一问题。只需要将盖章的时间地点以及印章所盖的文件的摘要用区块链记录下来，单纯的复制印章就可以被查出来，因为区块链中没有记录，且区块链的内容的修改需要付出很大的代价，只要超过罪犯能承受的上限，就不会有人愿意做，也就能确保印章真实可靠。盖章的记录都是公众可以查询到的，也确保其公开透明，容易获得公众的认可。

# 参考文献

[1]见证Linux的发展历史 天极网, 2004-9-28[2018-11-25], http://www.yesky.com/415/1858915.shtml

[2]回顾Linux20年光辉岁月 网络用户超过14亿. TechWeb网, 2011-08-17[2018-11-25]http://www.techweb.com.cn/world/2011-08-17/1081989.shtml

[3]国务院.《电子签名法》工业和信息化部, 2015-8-31[2018-11-25]<http://www.miit.gov.cn/n1146285/n1146352/n3054355/n3057254/n3057259/c3868973/content.html>

[4]国际化标准组织. PDF format become ISO standard[EB/OL].2010-1-1[2018-11-25].https://www.iso.org/news/2008/07/Ref1141.h